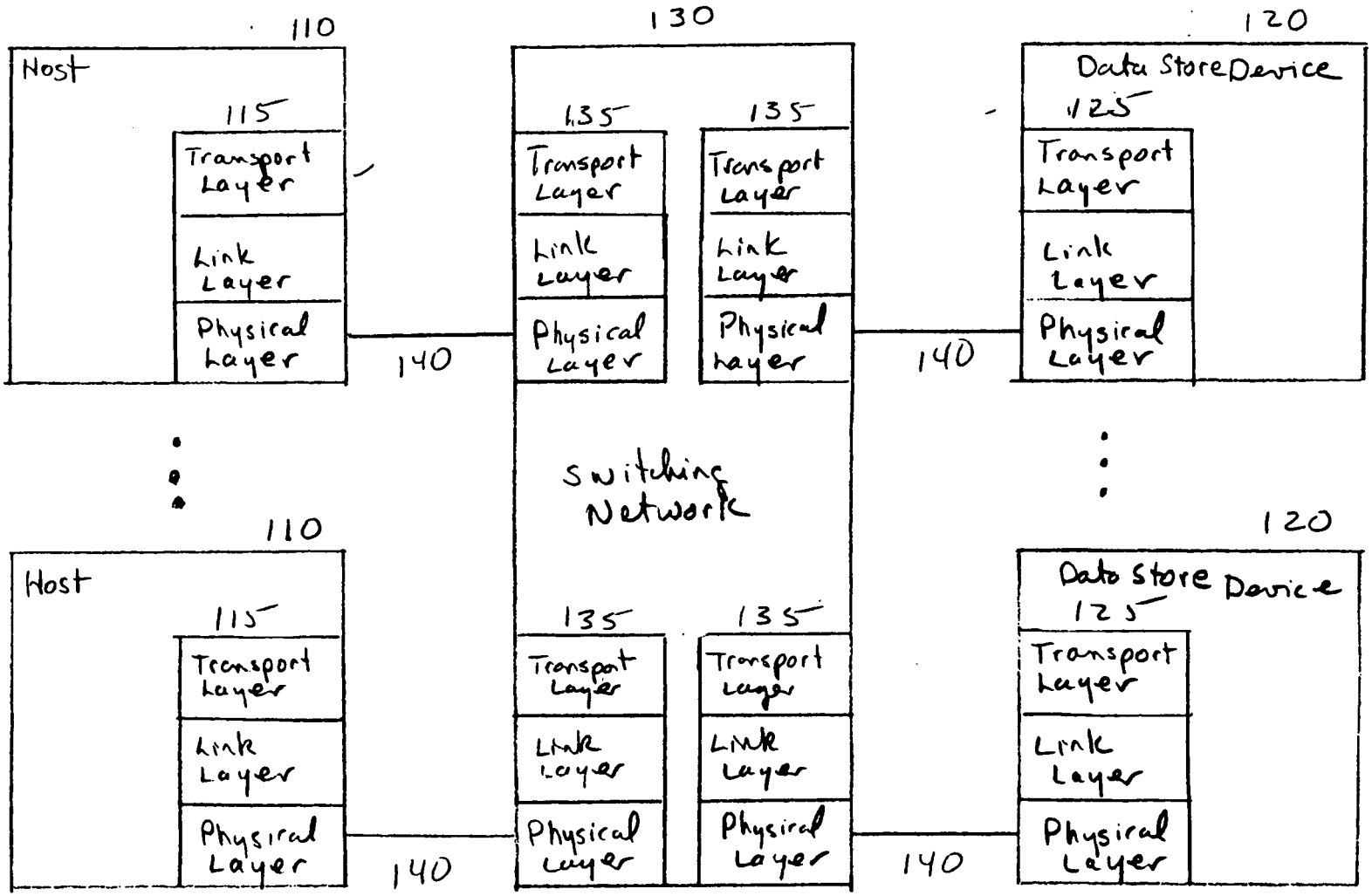

110
Host
115
Transport Layer
Link Layer
Physical Layer
140
130
135
Transport Layer
Link Layer
Physical Layer
135
Transport Layer
Link Layer
Physical Layer
140
120
Data Store Device
125
Transport Layer
Link Layer
Physical Layer
Switching Network
110
Host
115
Transport Layer
Link Layer
Physical Layer
140
135
Transport Layer
Link Layer
Physical Layer
135
Transport Layer
Link Layer
Physical Layer
140
120
Data Store Device
125
Transport Layer
Link Layer
Physical Layer

Fig 1

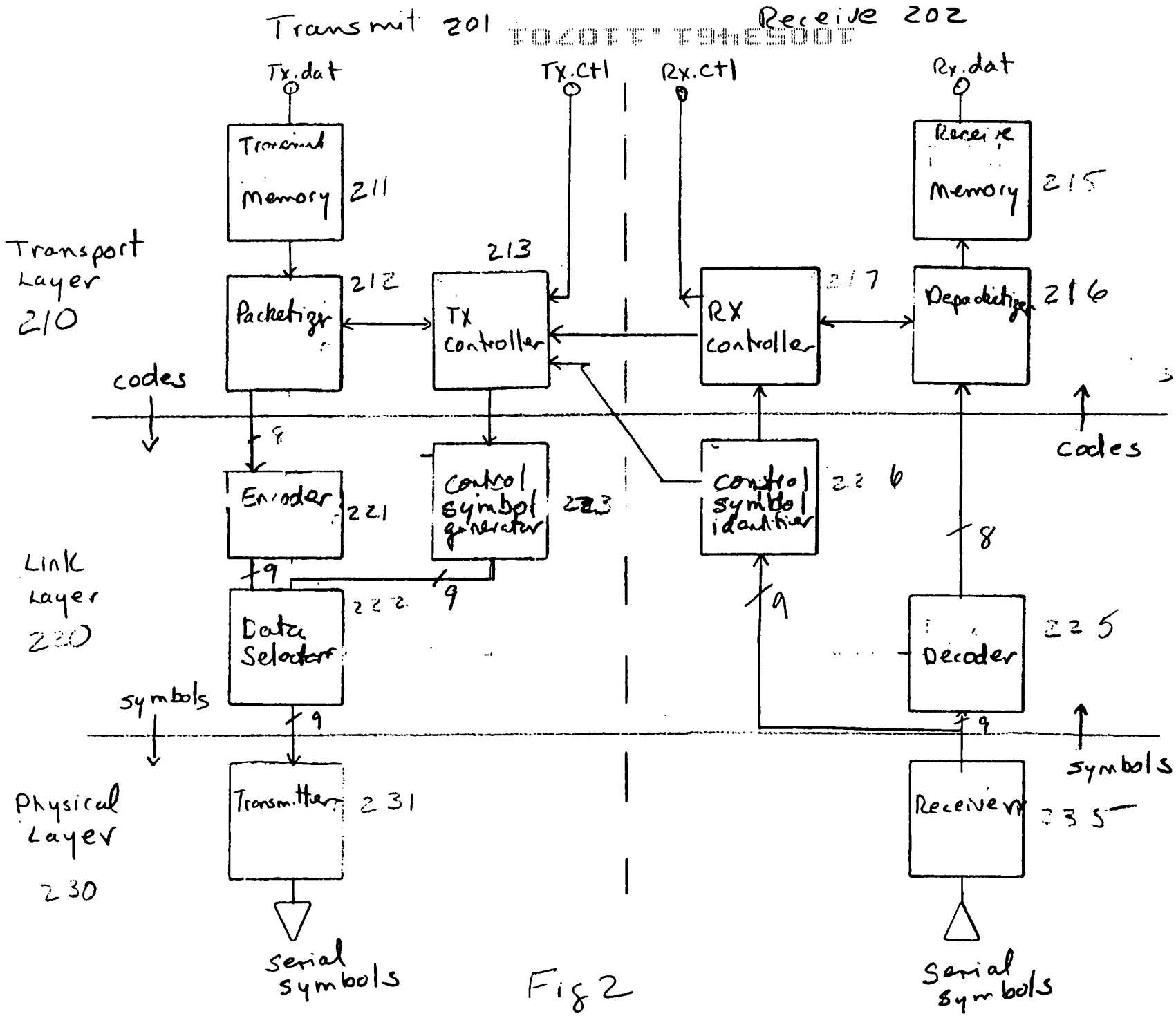

Transmit 201
Receive 202
Tx.dat
Tx.Ctl
Rx.Ctl
Rx.dat
Transmit memory 211
Packetizer 212
TX controller 213
RX controller 217
Receive memory 215
Depacketizer 216
Transport Layer 210
codes
8
Encoder 221
control symbol generator 223
control symbol identifier 226
9
Data Selector 222
Decoder 225
Link Layer 220
symbols
Transmitter 231
Receiver 235
Physical Layer 230
serial symbols
serial symbols

Fig 2

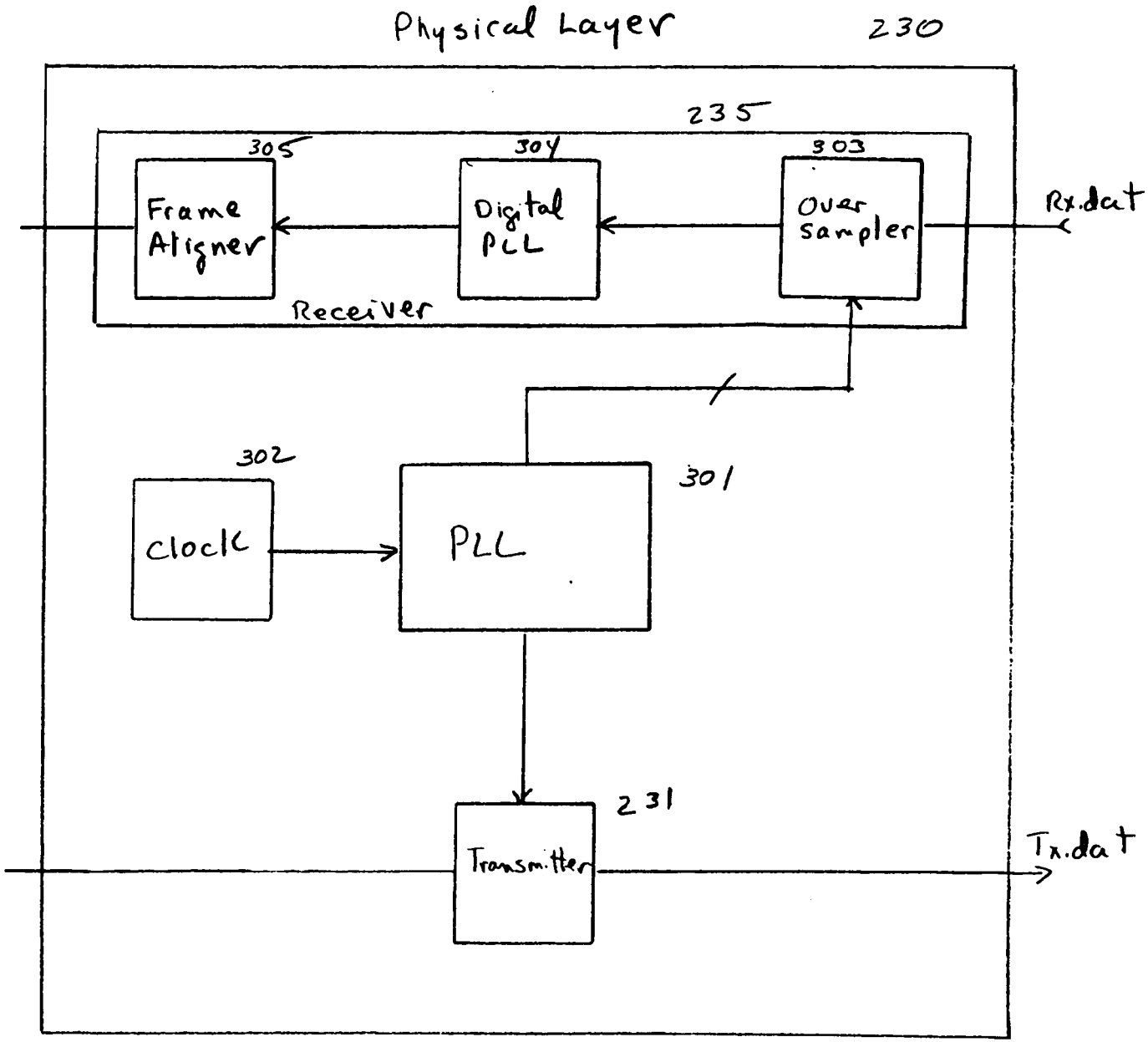


Fig 3

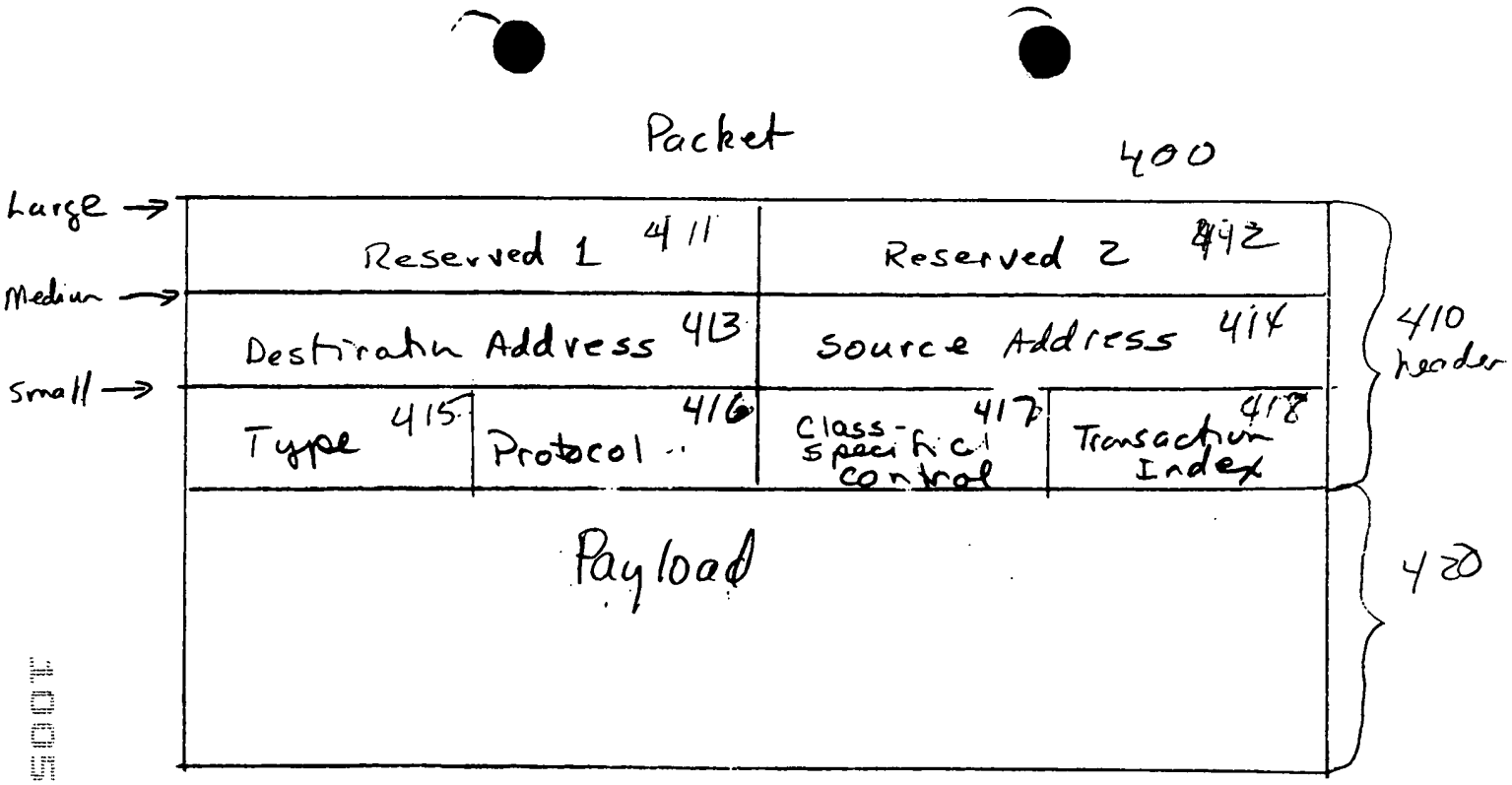

Packet
400
Large →
Medium →
Small →
Reserved 1
411
Reserved 2
412
Destination Address
413
Source Address
414
Type
415
Protocol
416
Class-specific control
417
Transaction Index
418
410 header
Payload
420

Fig 4

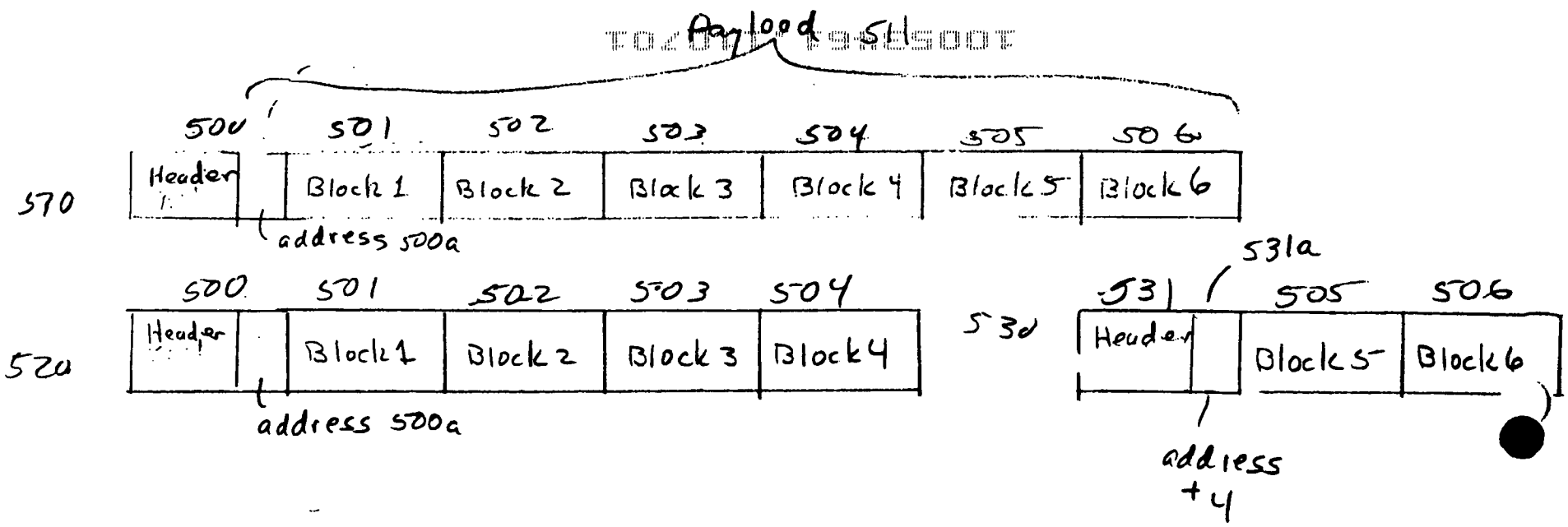

Payload
500
501
502
503
504
505
506
570
Header
Block 1
Block 2
Block 3
Block 4
Block 5
Block 6
address 500a
500
501
502
503
504
520
Header
Block 1
Block 2
Block 3
Block 4
address 500a
531a
531
505
506
530
Header
Block 5
Block 6
address
+4

Fig 5

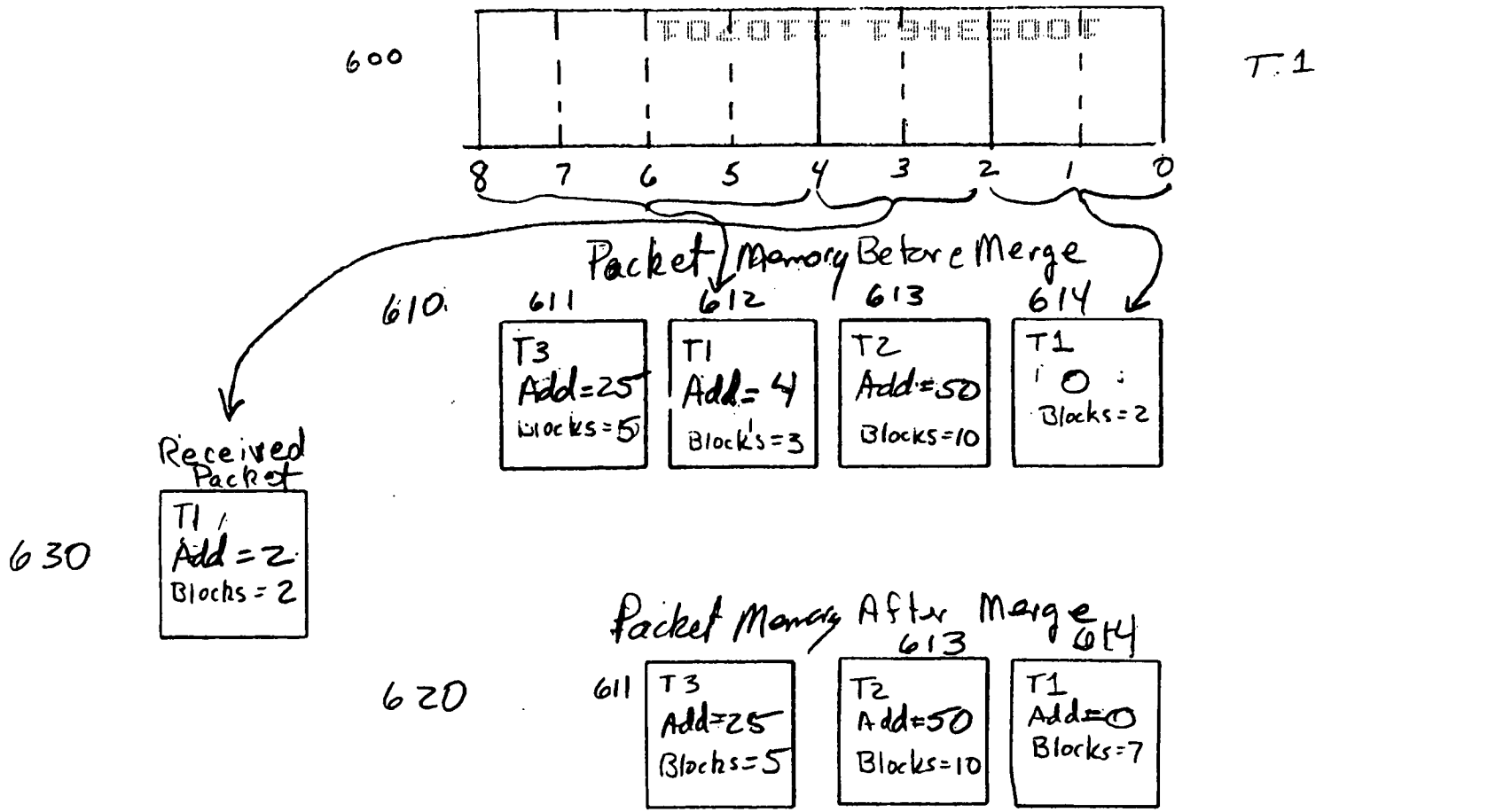

600
T1
8 7 6 5 4 3 2 1 0
Packet Memory Before Merge
610
611
612
613
614
T3
Add=25
Blocks=5
T1
Add=4
Blocks=3
T2
Add=50
Blocks=10
T1
Add=0
Blocks=2
Received Packet
630
T1
Add=2
Blocks=2
Packet Memory After Merge
620
611
613
614
T3
Add=25
Blocks=5
T2
Add=50
Blocks=10
T1
Add=0
Blocks=7

Fig 6

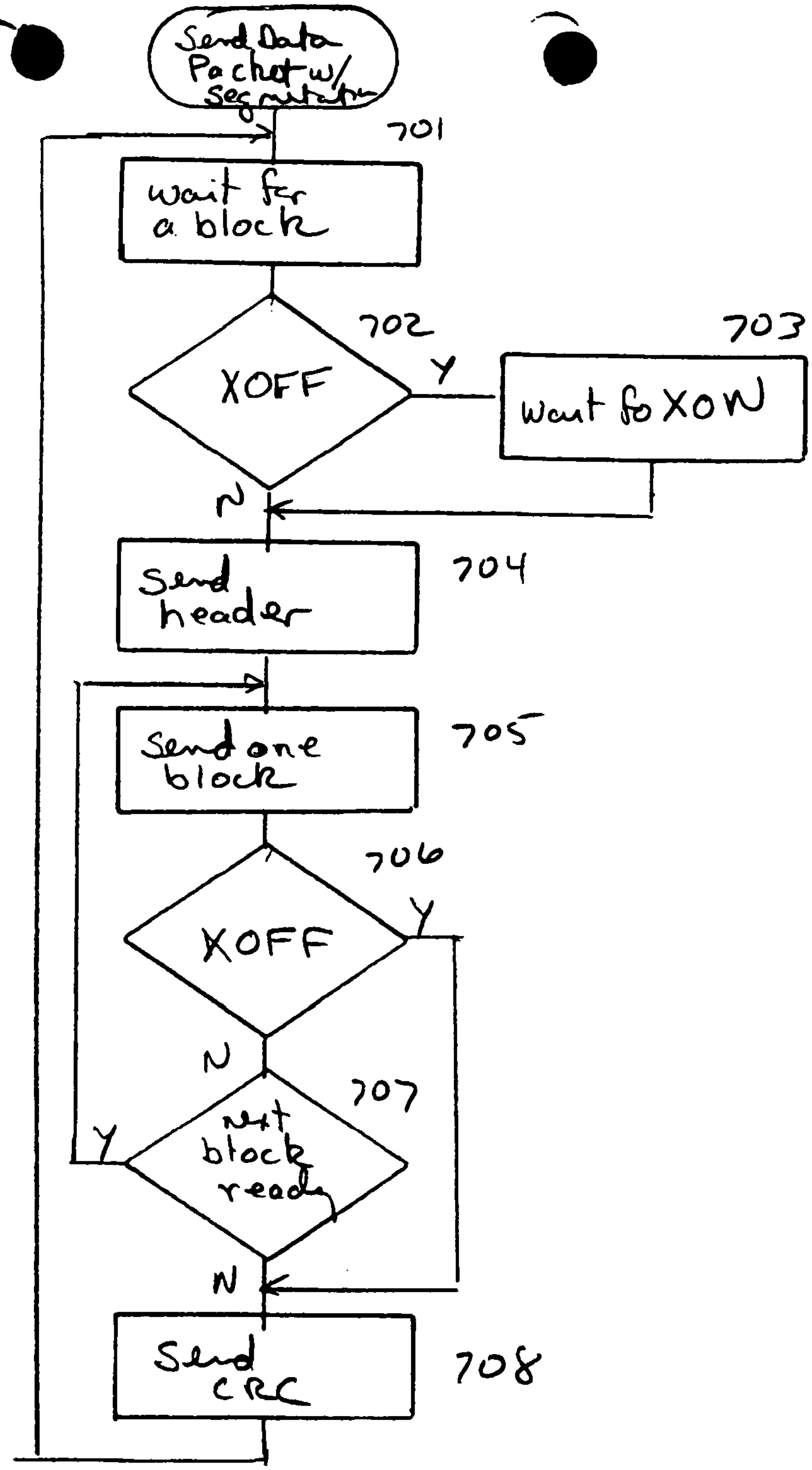


Fig 7

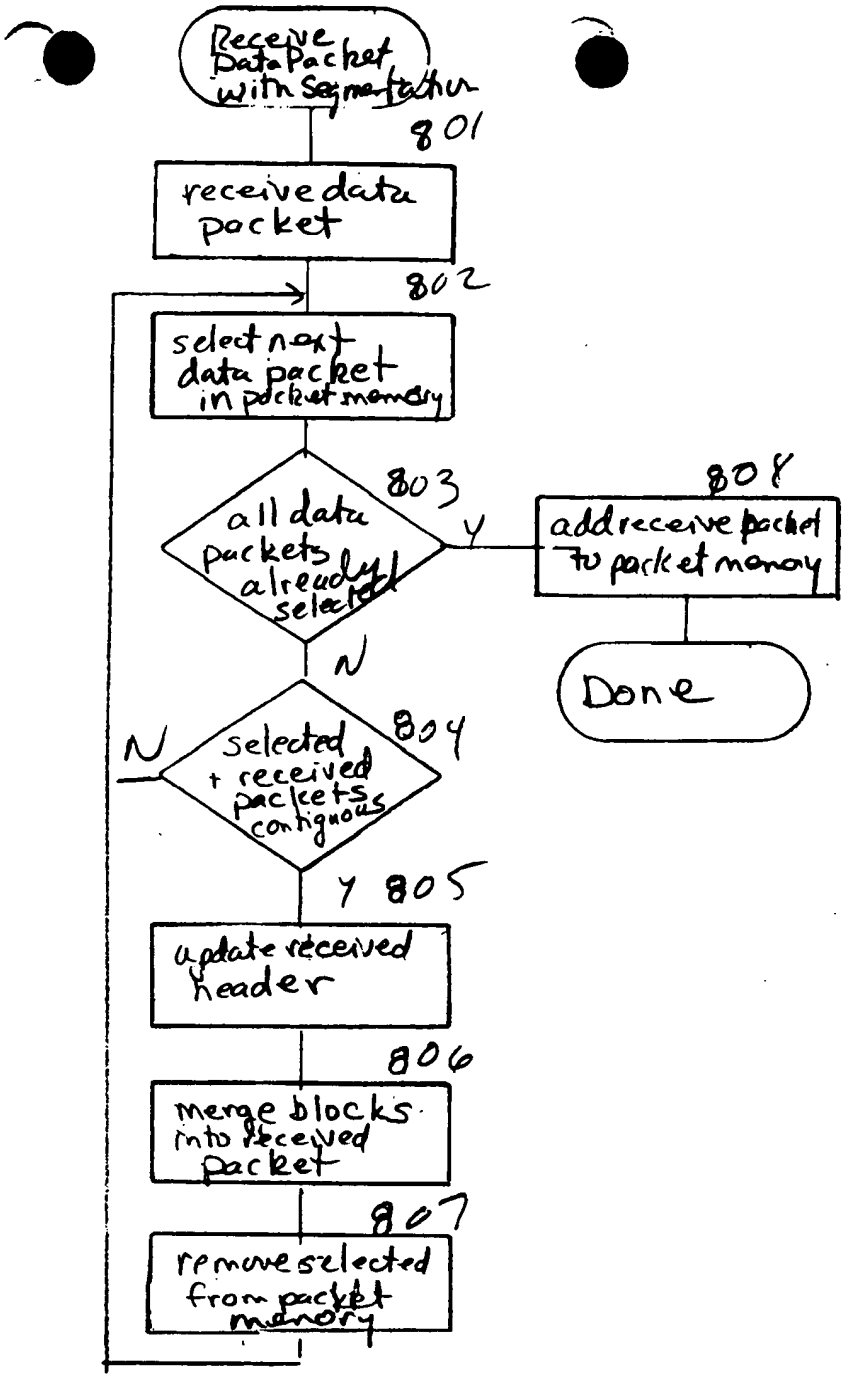
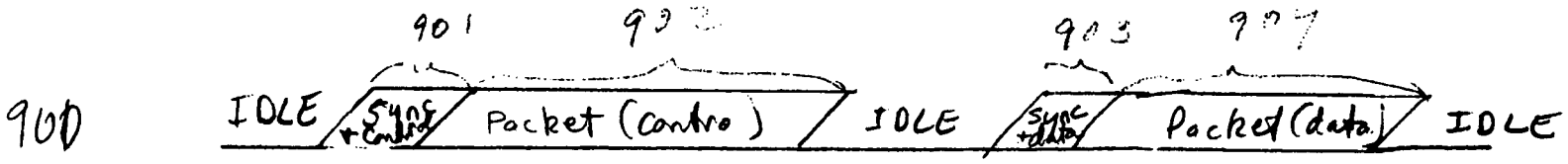

Receive Data Packet with Segmentation
801
receive data packet
802
select next data packet in packet memory
803
all data packets already selected
Y
808
add receive packet to packet memory
Done
N
N
selected + received packets contiguous
804
Y 805
update received header
806
merge blocks into received packet
807
remove selected from packet memory

Fig. 8



Sync + packet type

Fig 9A

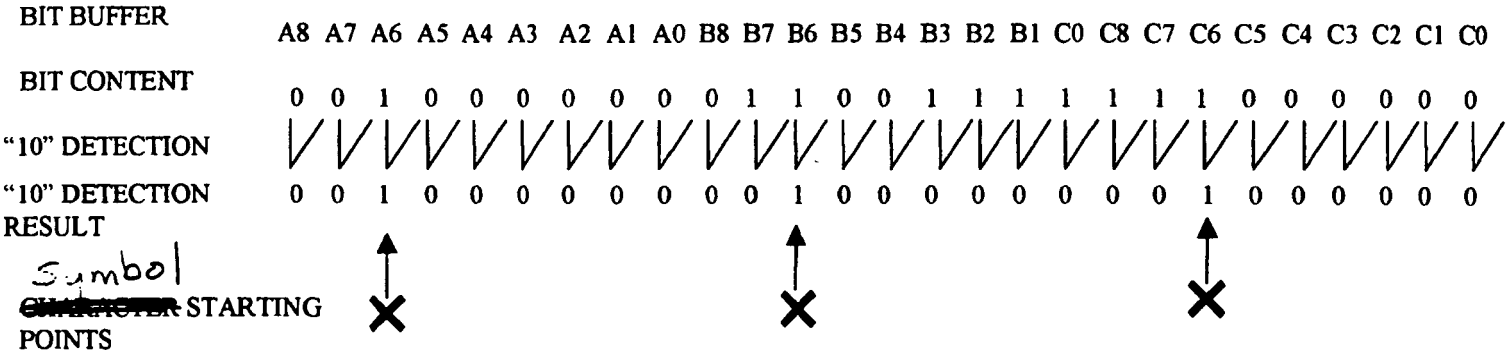


FIG.10

Fig 9B

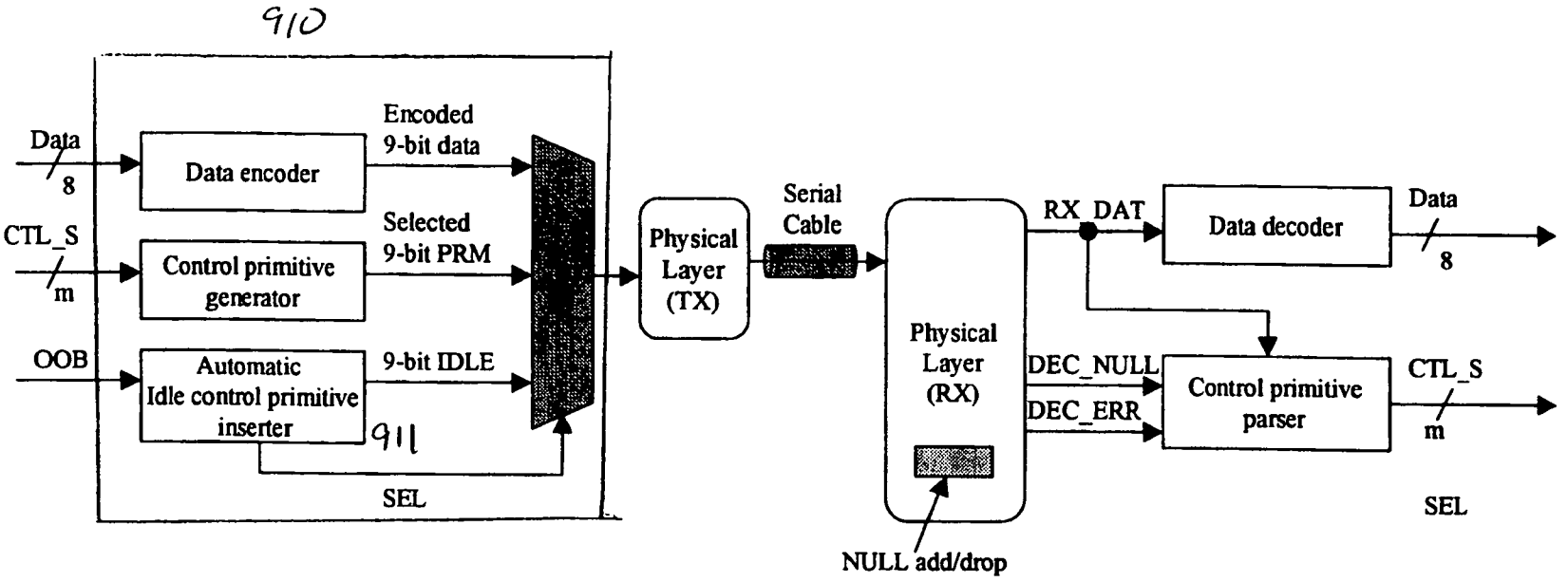


Fig. 9C

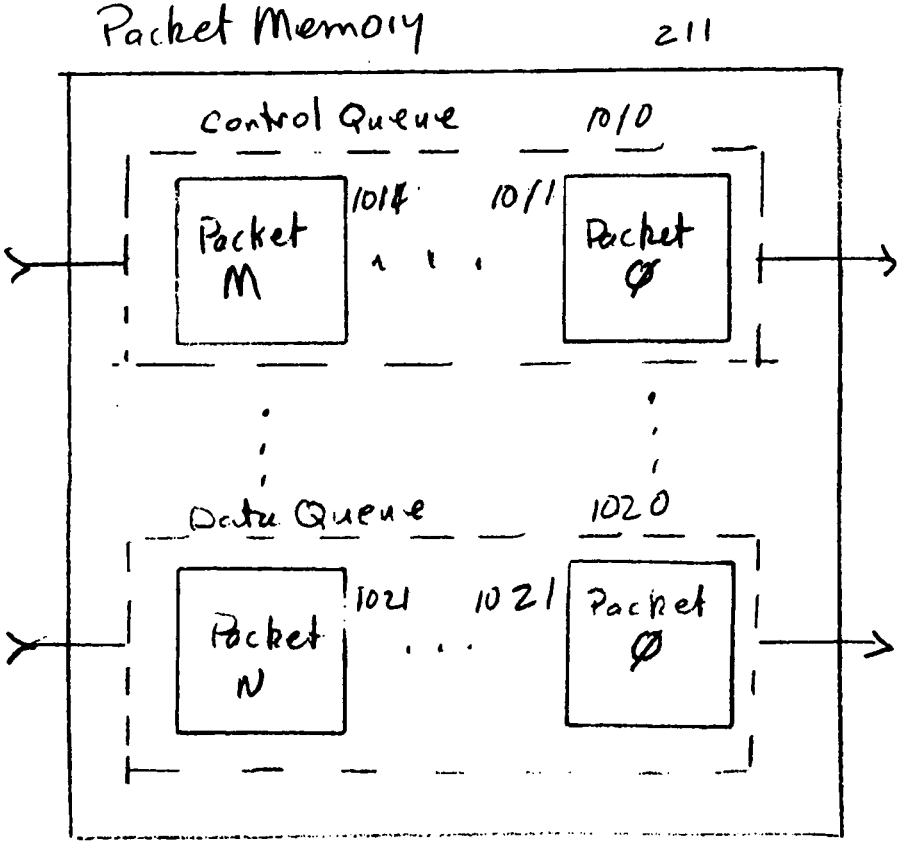


Fig 10

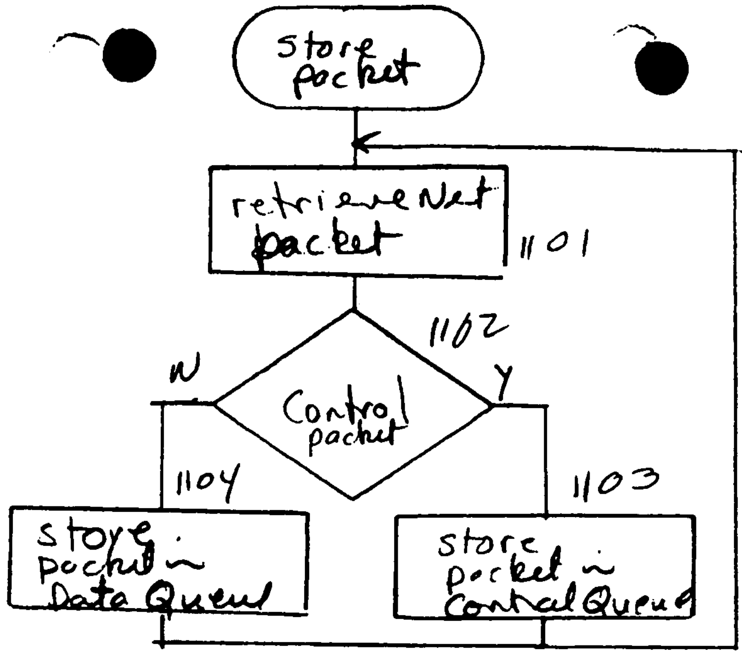


Fig 11

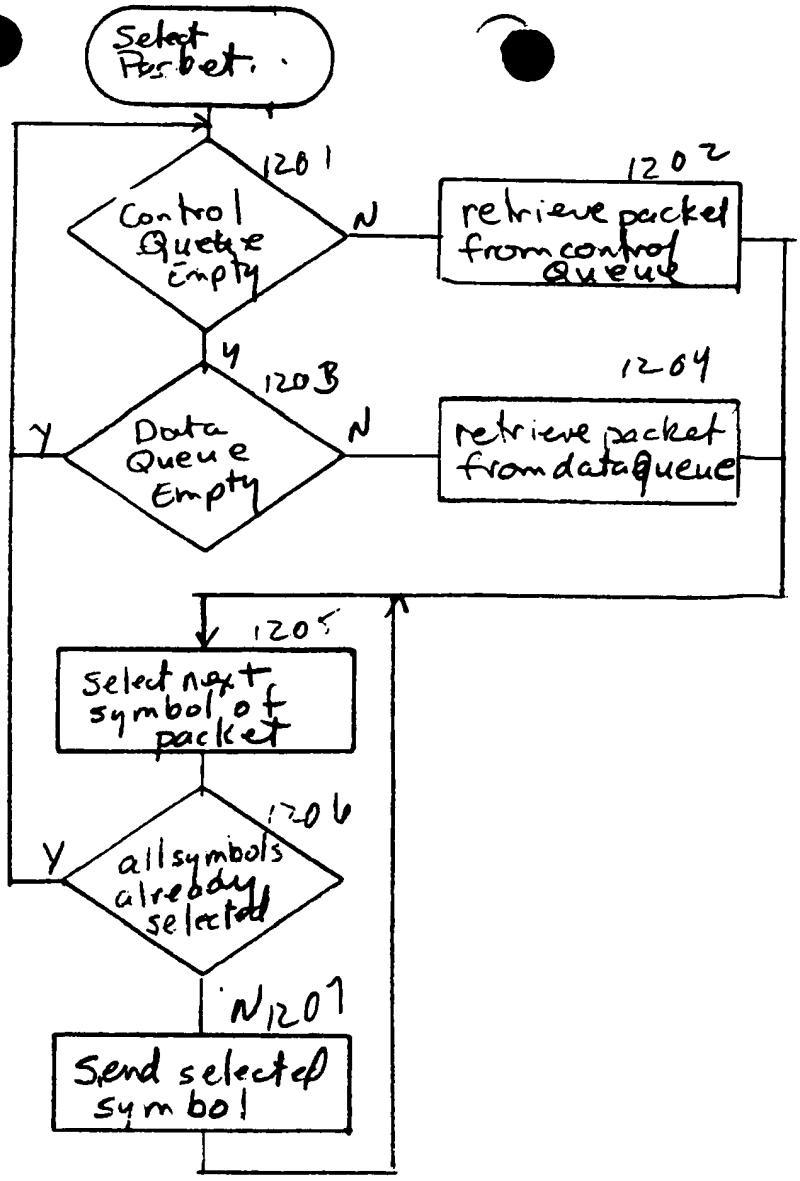

Select Packet
1201
Control Queue Empty
N
1202
retrieve packet from control queue
Y
1203
Data Queue Empty
Y
N
1204
retrieve packet from data queue
1205
Select next symbol of packet
1206
all symbols already selected
Y
N 1207
Send selected symbol

Fig 12

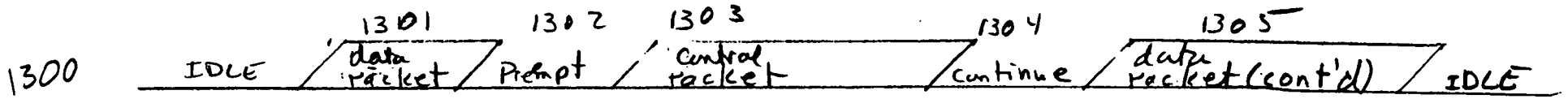


Fig 13

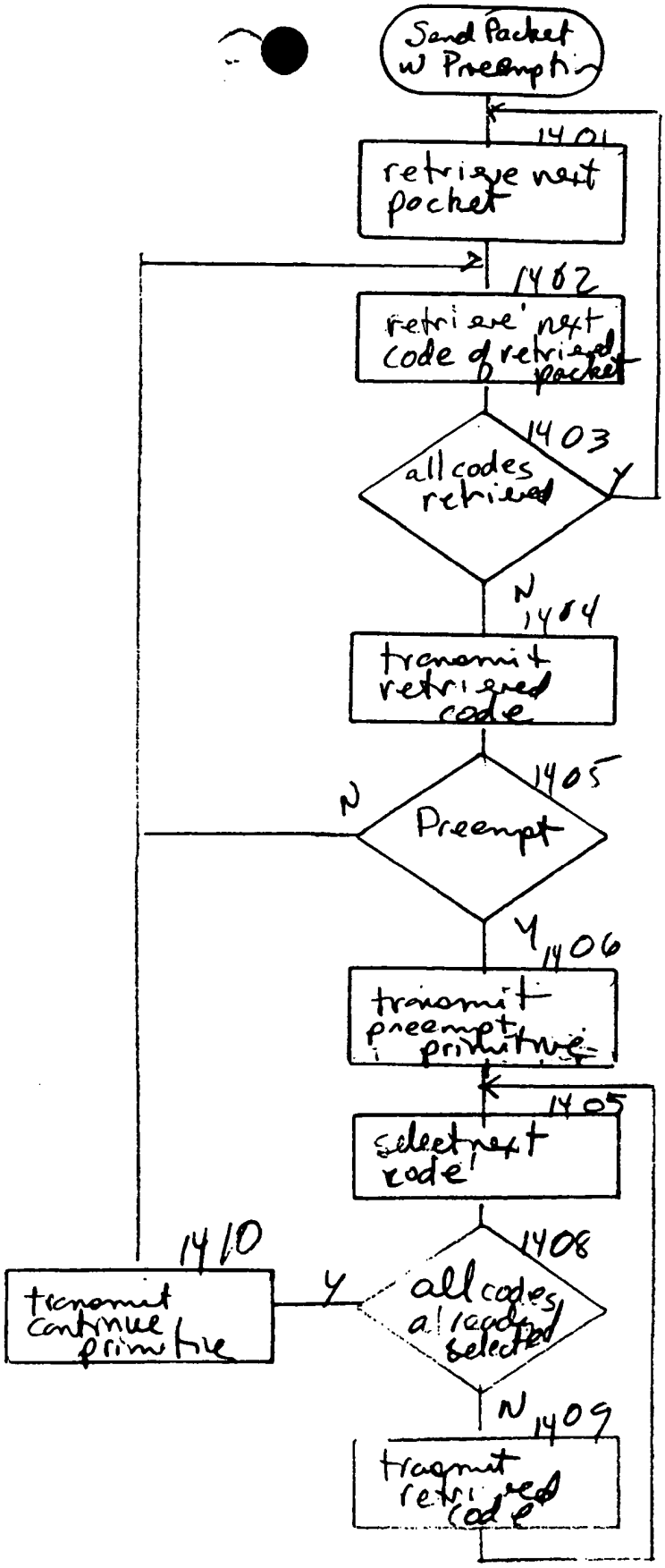


Fig 14

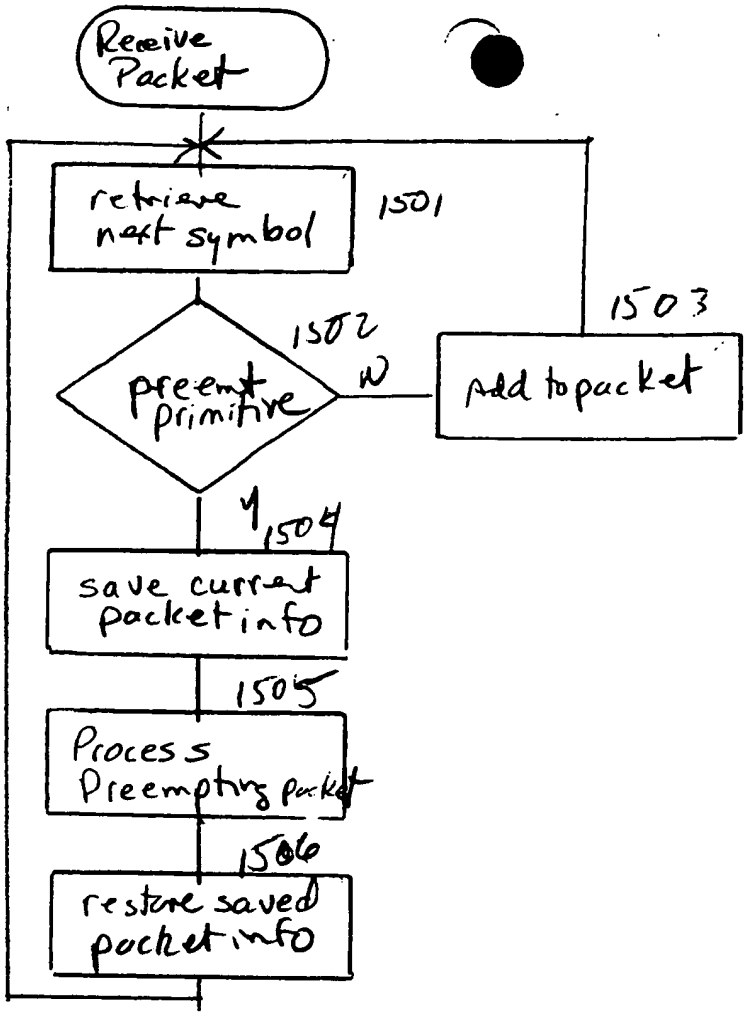

Receive Packet
retrieve next symbol
1501
preempt primitive
1502
N
Add to packet
1503
Y
1504
save current packet info
1505
Process Preempting packet
1506
restore saved packet info

Fig 15

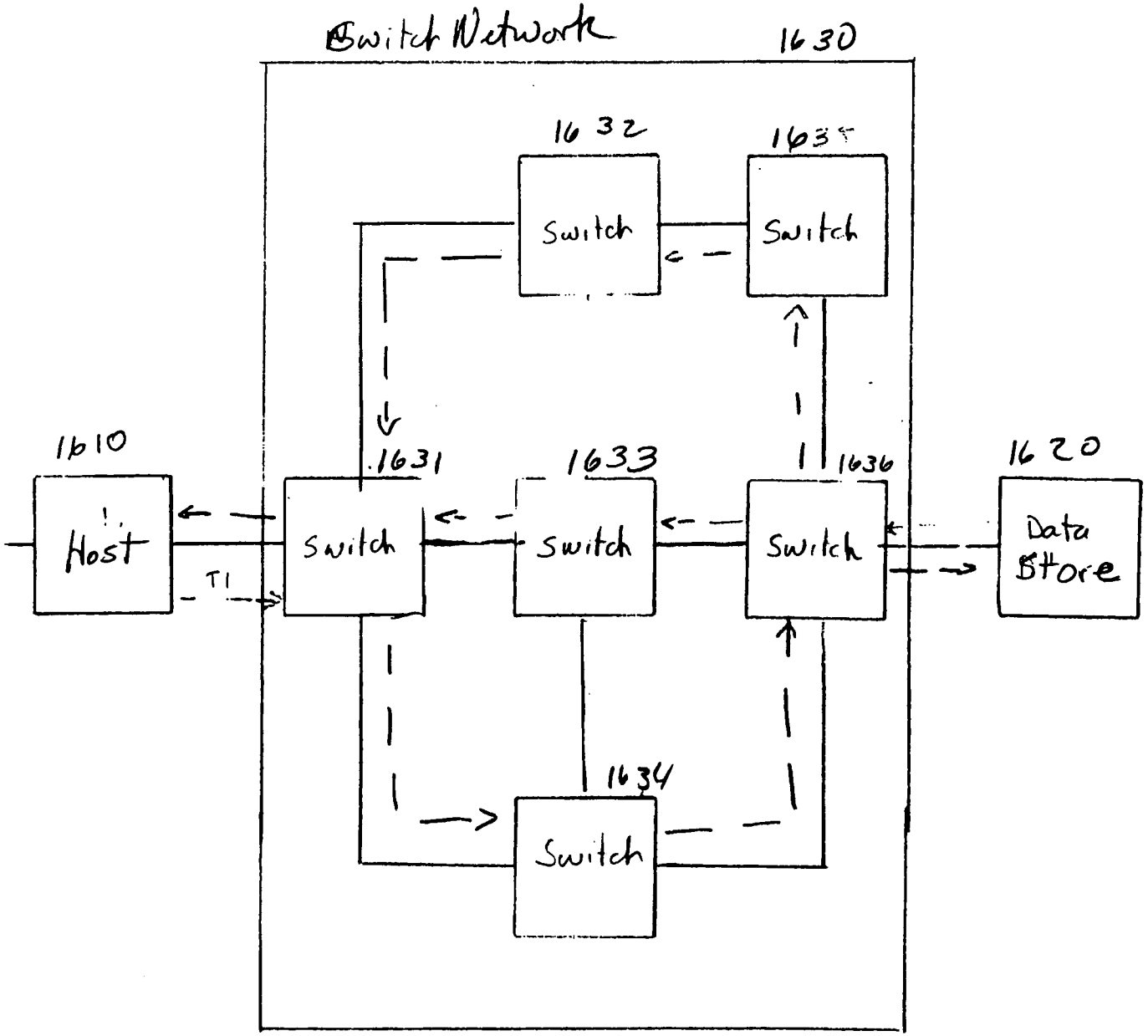


Fig 16

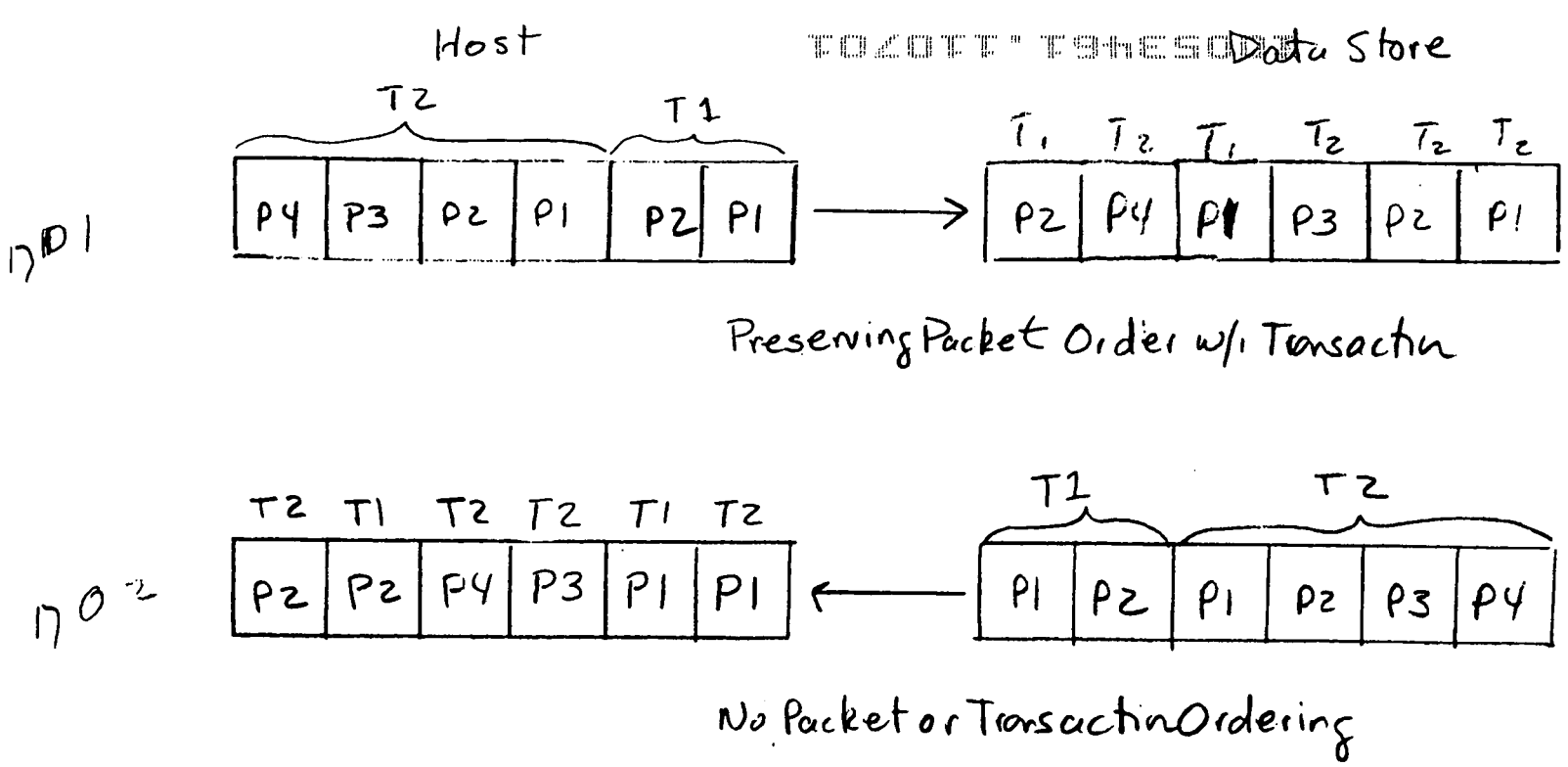

Host
Data Store
T2
T1
P4 P3 P2 P1 P2 P1
T1 T2 T1 T2 T2 T2
P2 P4 P1 P3 P2 P1
Preserving Packet Order w/ Transaction
T2 T1 T2 T2 T1 T2
P2 P2 P4 P3 P1 P1
T1
T2
P1 P2 P1 P2 P3 P4
No Packet or Transaction Ordering

Fig 17

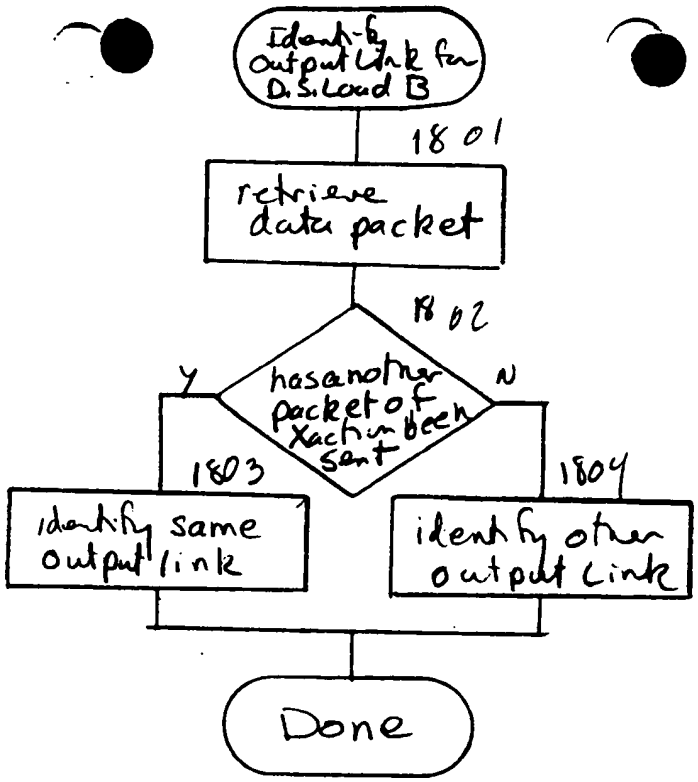


Fig 18

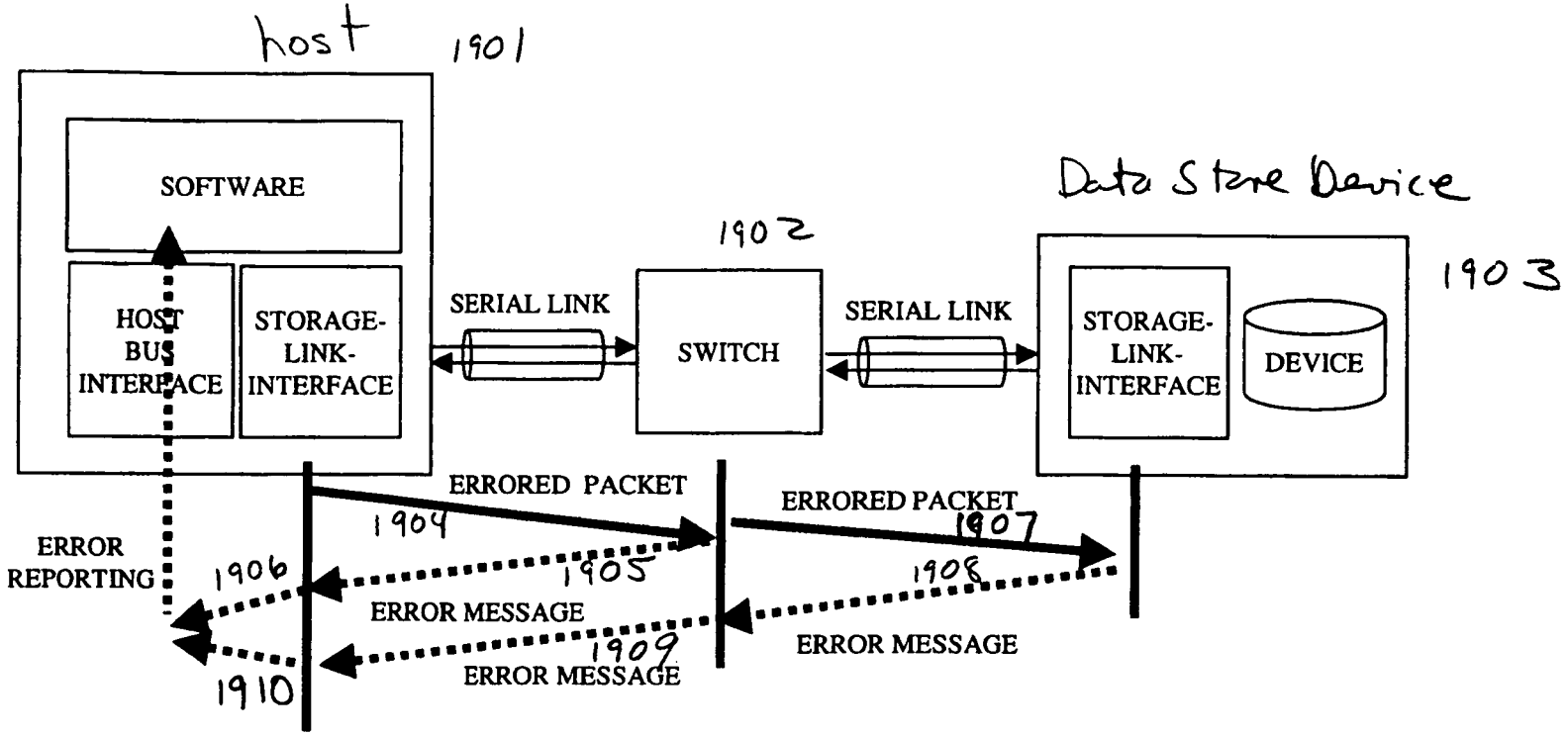


Fig 19A

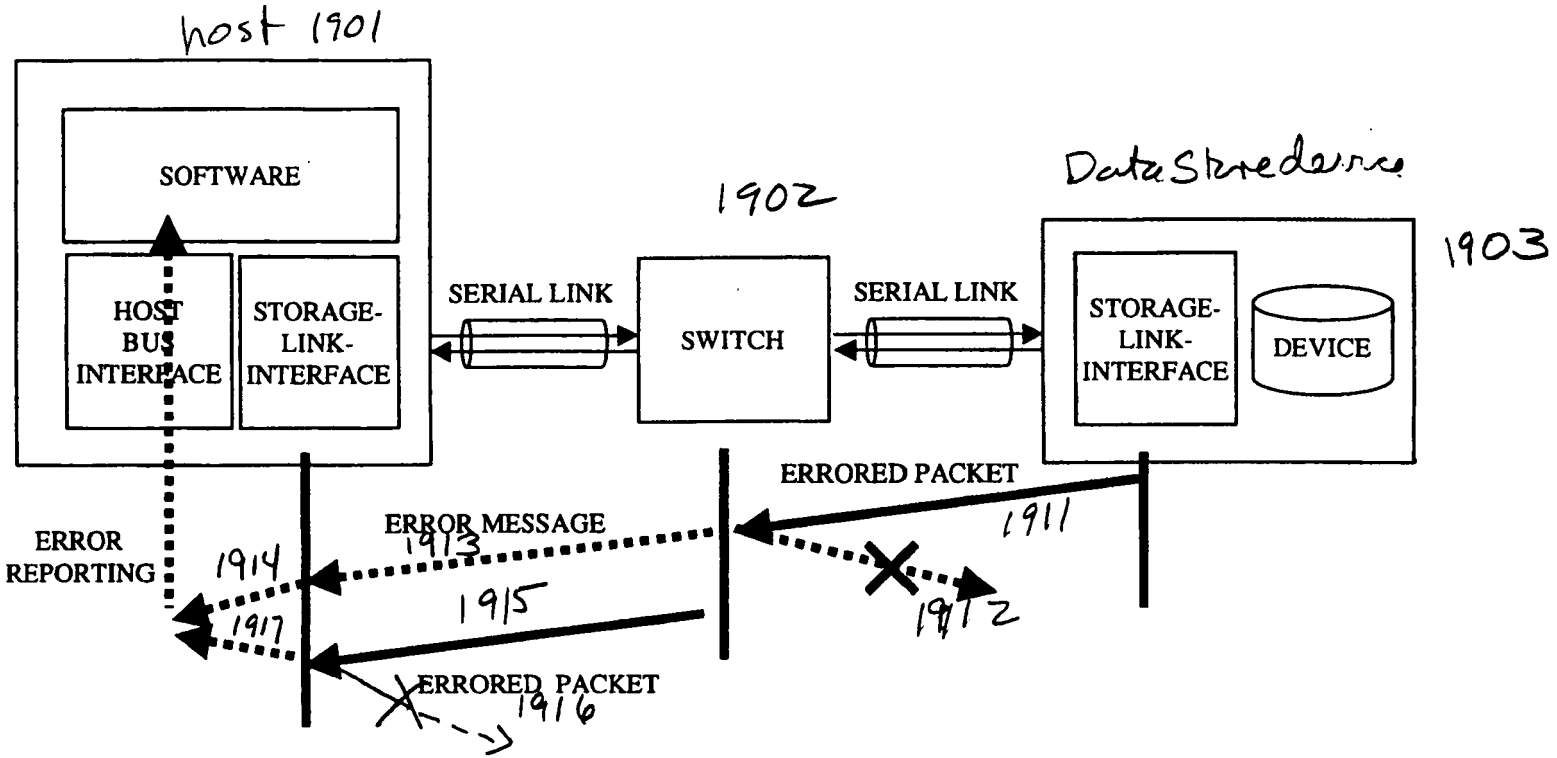
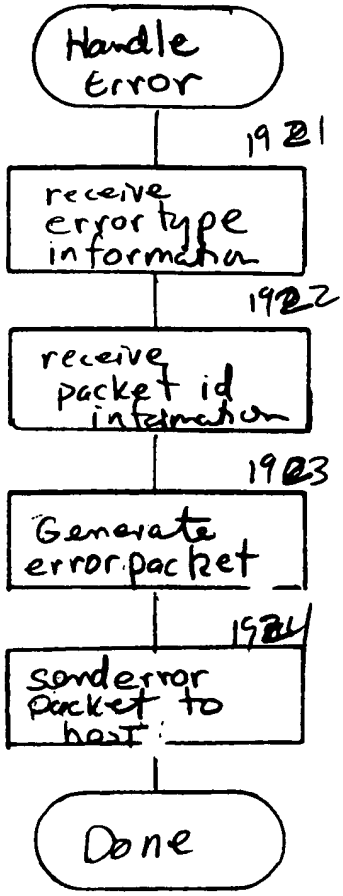


Fig 19B

FIG20


Handle Error
1921
receive error type information
1922
receive packet id information
1923
Generate error packet
1924
send error packet to host
Done

19C

8b Code	9 bit symbol
0000 0000	101010101
0000 0001	101010100
0000 0010	101010111
⋮	
0101 0101	001010101
⋮	
0111 0110	001110110
0111 0111	100100010
⋮	
1111 1111	110101010

Fig 20

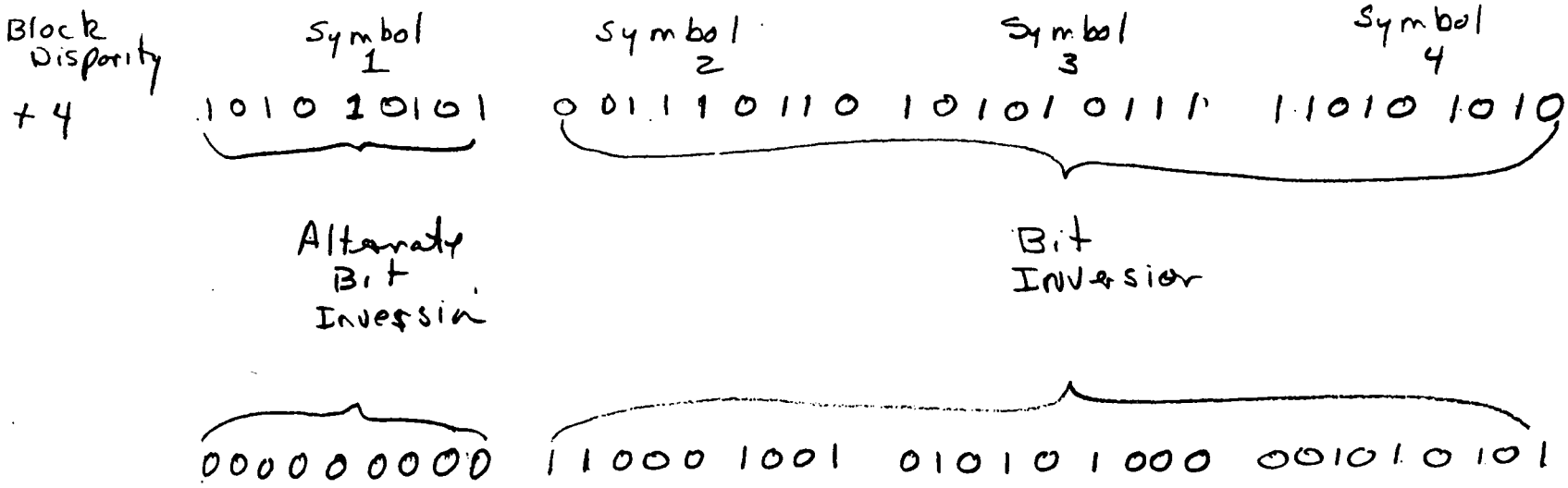


Fig 21A

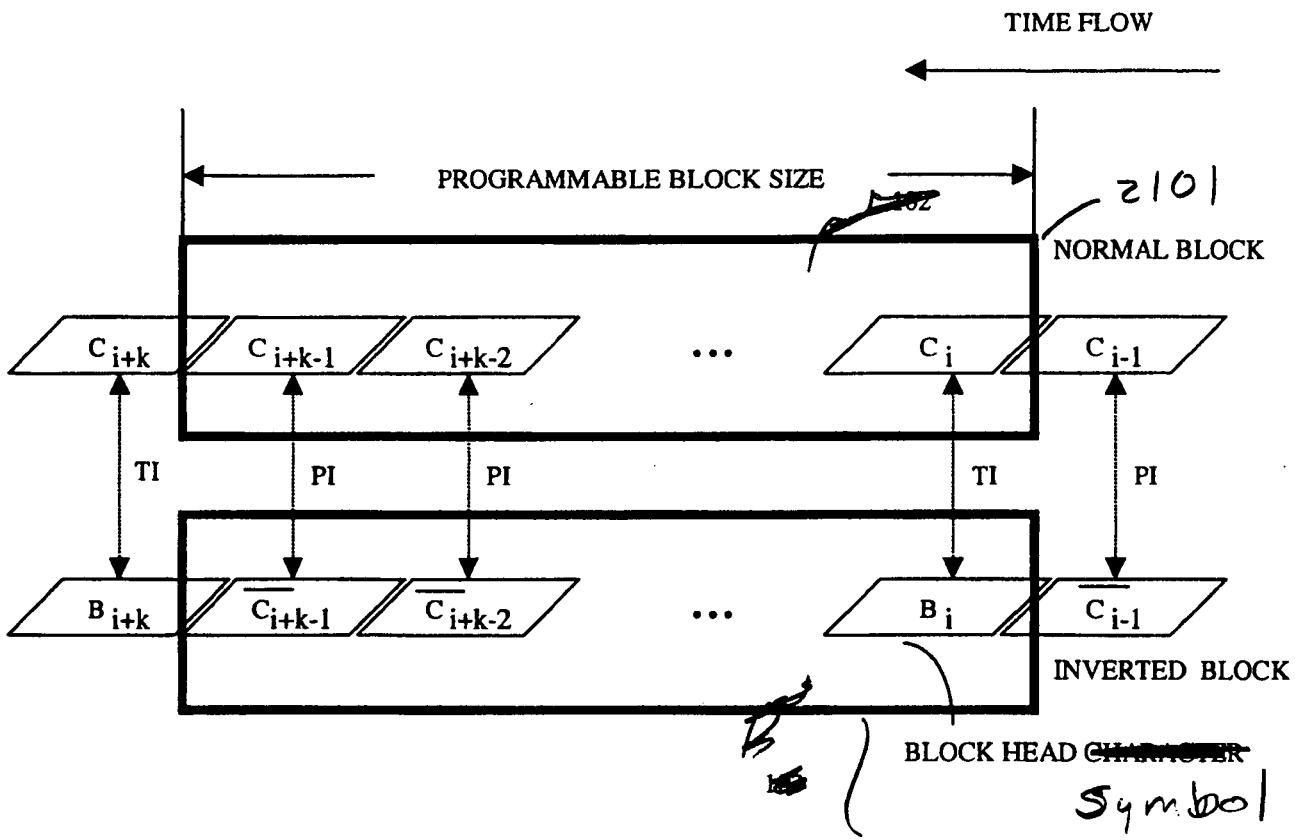

TIME FLOW
PROGRAMMABLE BLOCK SIZE
2101
NORMAL BLOCK
C i+k
C i+k-1
C i+k-2
...
C i
C i-1
TI
PI
PI
TI
PI
B i+k
C i+k-1
C i+k-2
...
B i
C i-1
INVERTED BLOCK
BLOCK HEAD
Symbol
2102

Fig 21B

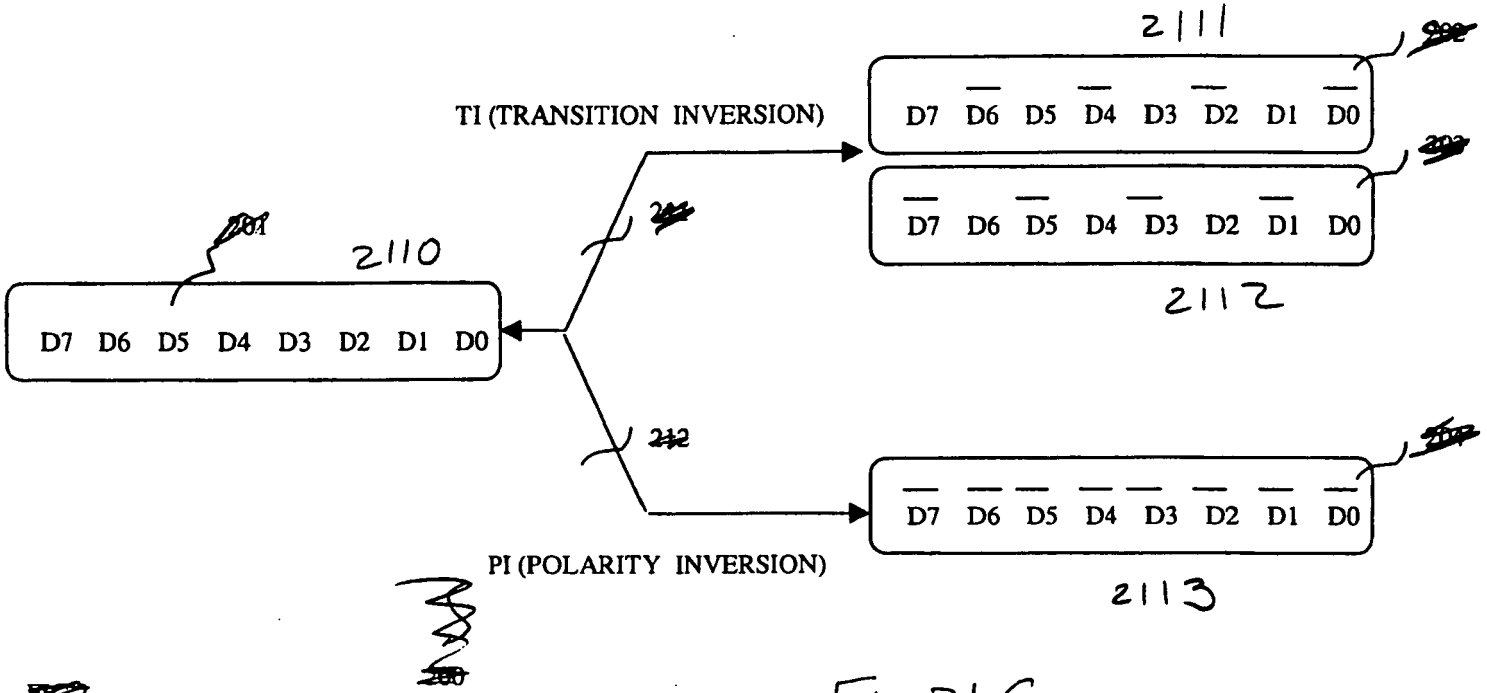


Fig 21C

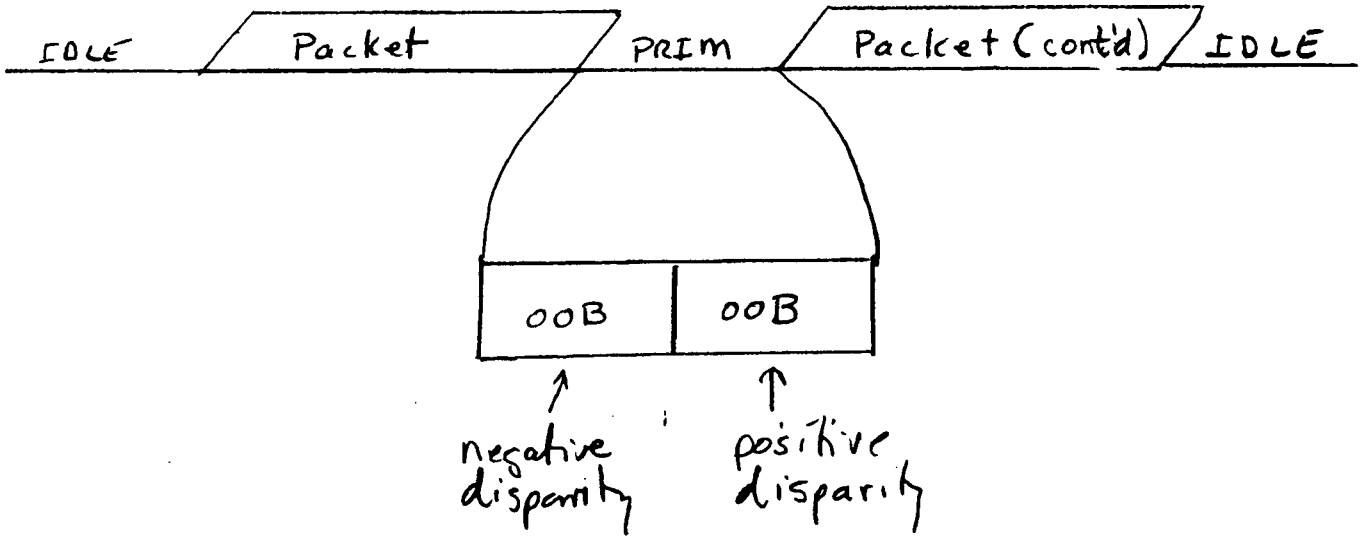


Fig 22

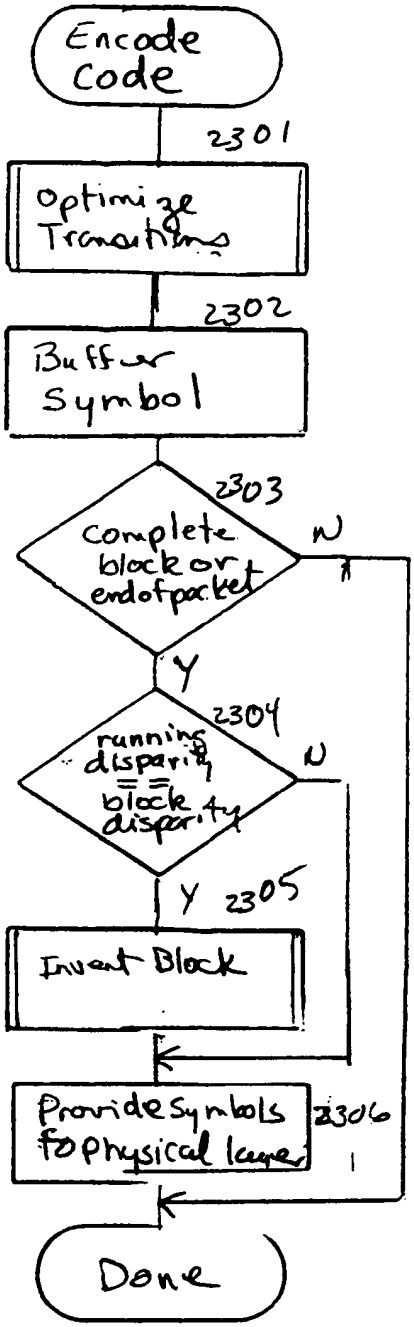

Encode Code
2301
Optimize Transitions
2302
Buffer Symbol
2303
Complete block or endofpacket
N
Y
2304
running disparity == block disparity
N
Y
2305
Invert Block
Provide Symbols fo Physical layer
2306
Done

Fig 23

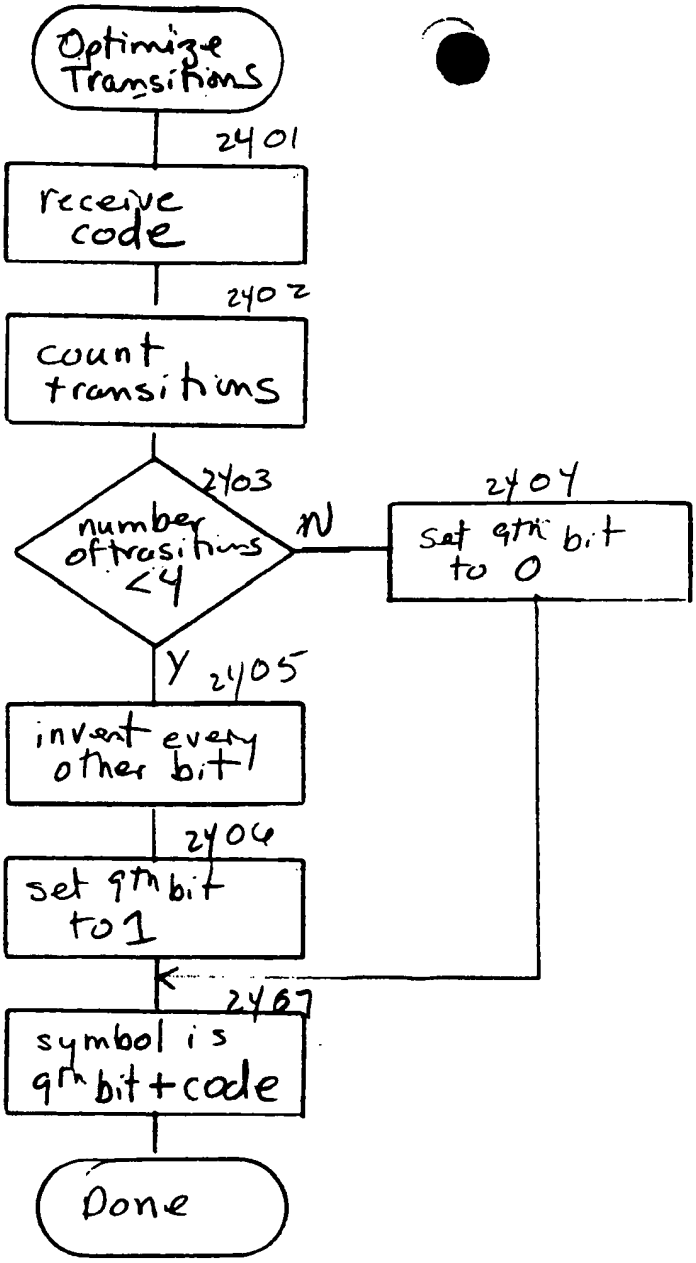

Optimize Transitions
2401
receive code
2402
count transitions
2403
number of trasitions <4
N
2404
set qth bit to 0
Y
2405
invert every other bit
2406
set qth bit to 1
2407
symbol is qth bit + code
Done

Fig 24

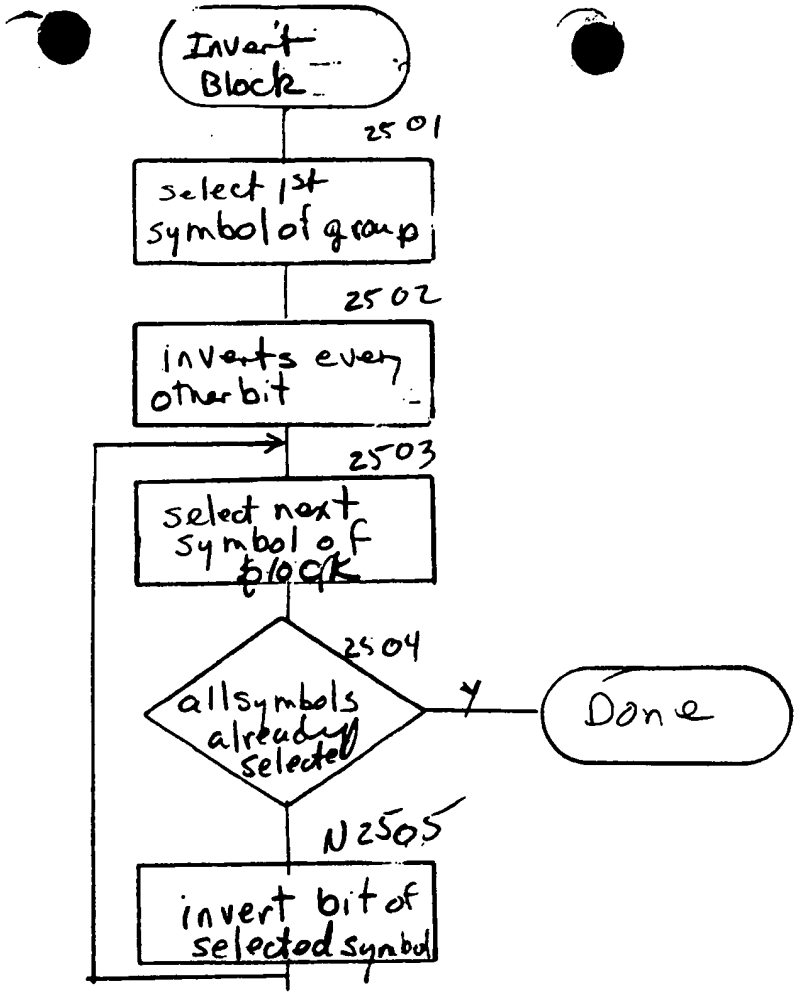

Invert Block
2501
select 1st symbol of group
2502
inverts every other bit
2503
select next symbol of block
2504
all symbols already selected
Y
Done
N
2505
invert bit of selected symbol

Fig 25

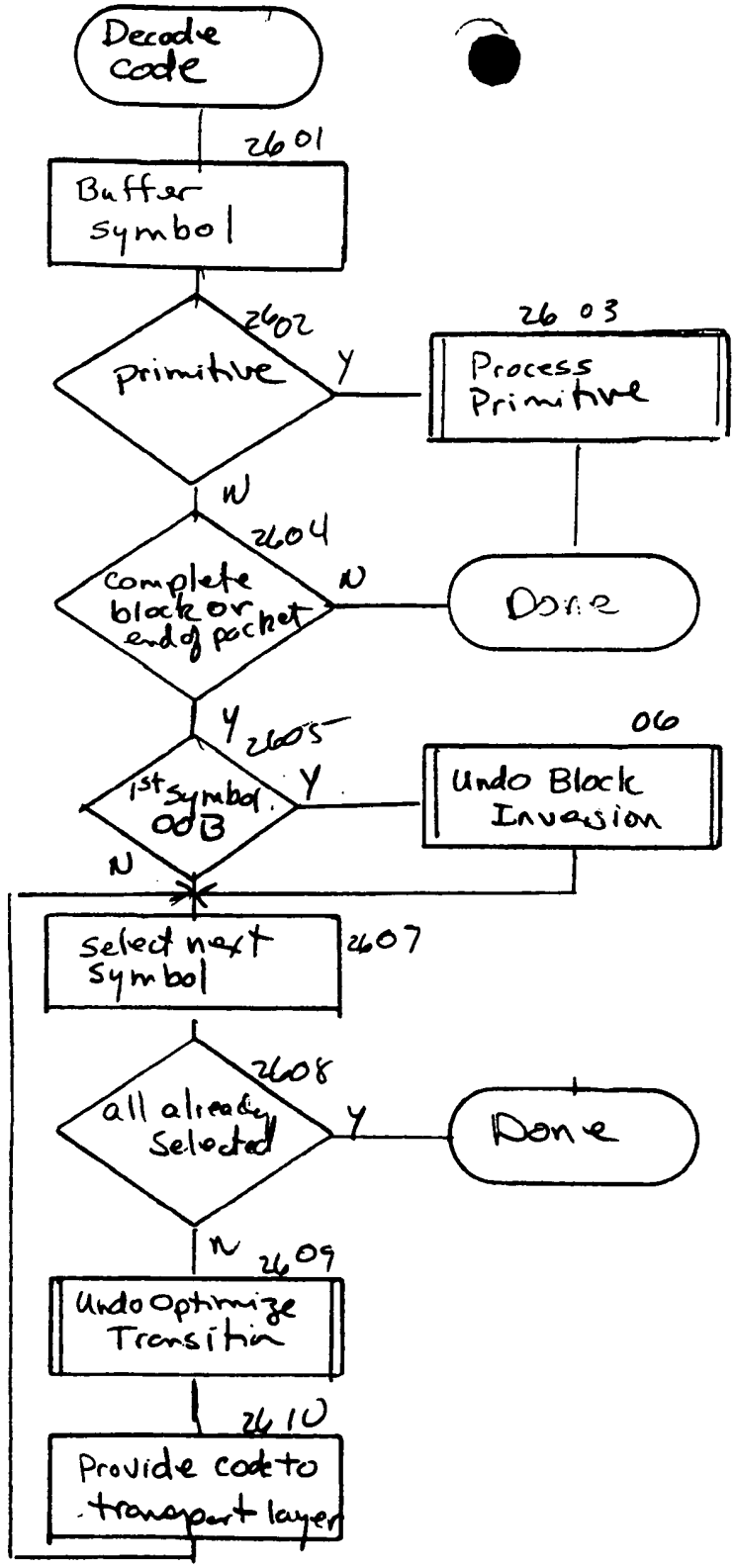

Decode code
2601
Buffer symbol
2602
Primitive
Y
2603
Process Primitive
N
2604
Complete block or end of packet
N
Done
Y
2605
1st symbol 00B
Y
06
Undo Block Invasion
N
Select next symbol
2607
2608
all already Selected
Y
Done
N
2609
Undo Optimize Transition
2610
Provide code to transport layer

Fig 26

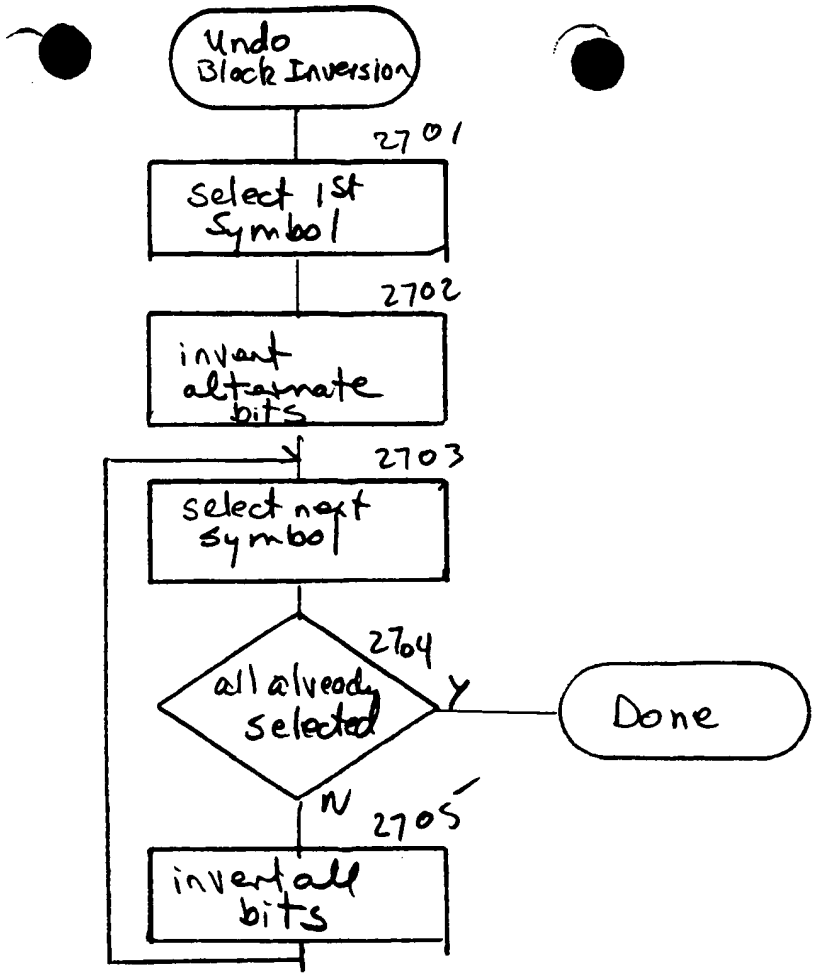

Undo Block Inversion
2701
Select 1st Symbol
2702
invert alternate bits
2703
select next symbol
2704
all already selected
Y
Done
N
2705
invert all bits

Fig 27

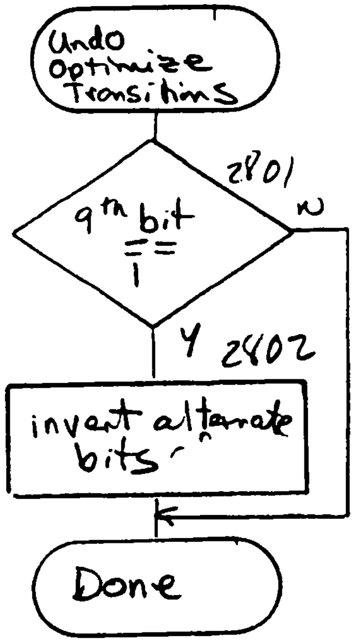


Fig 28

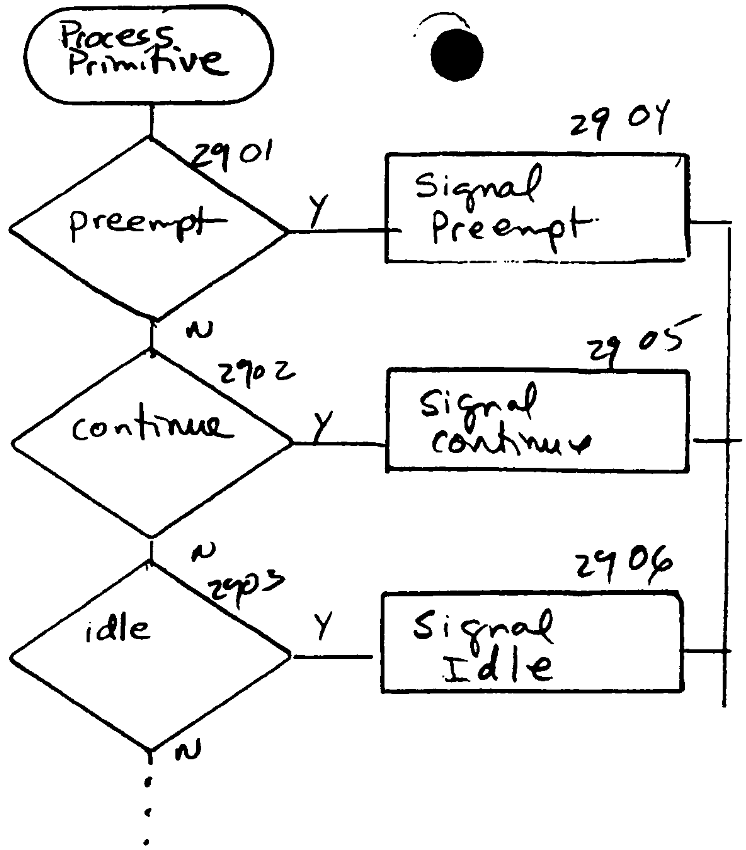


Fig 29

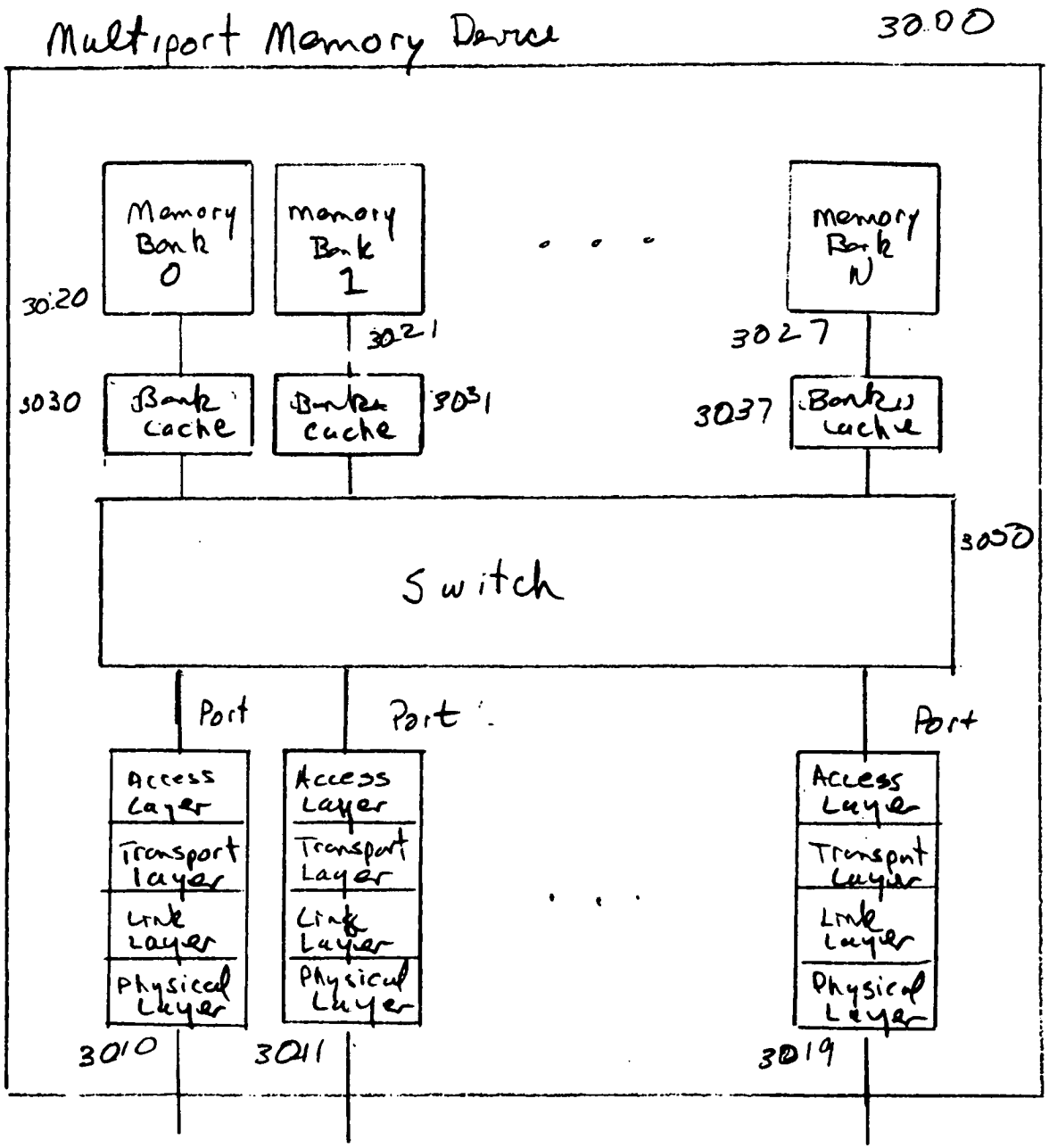


Fig 30

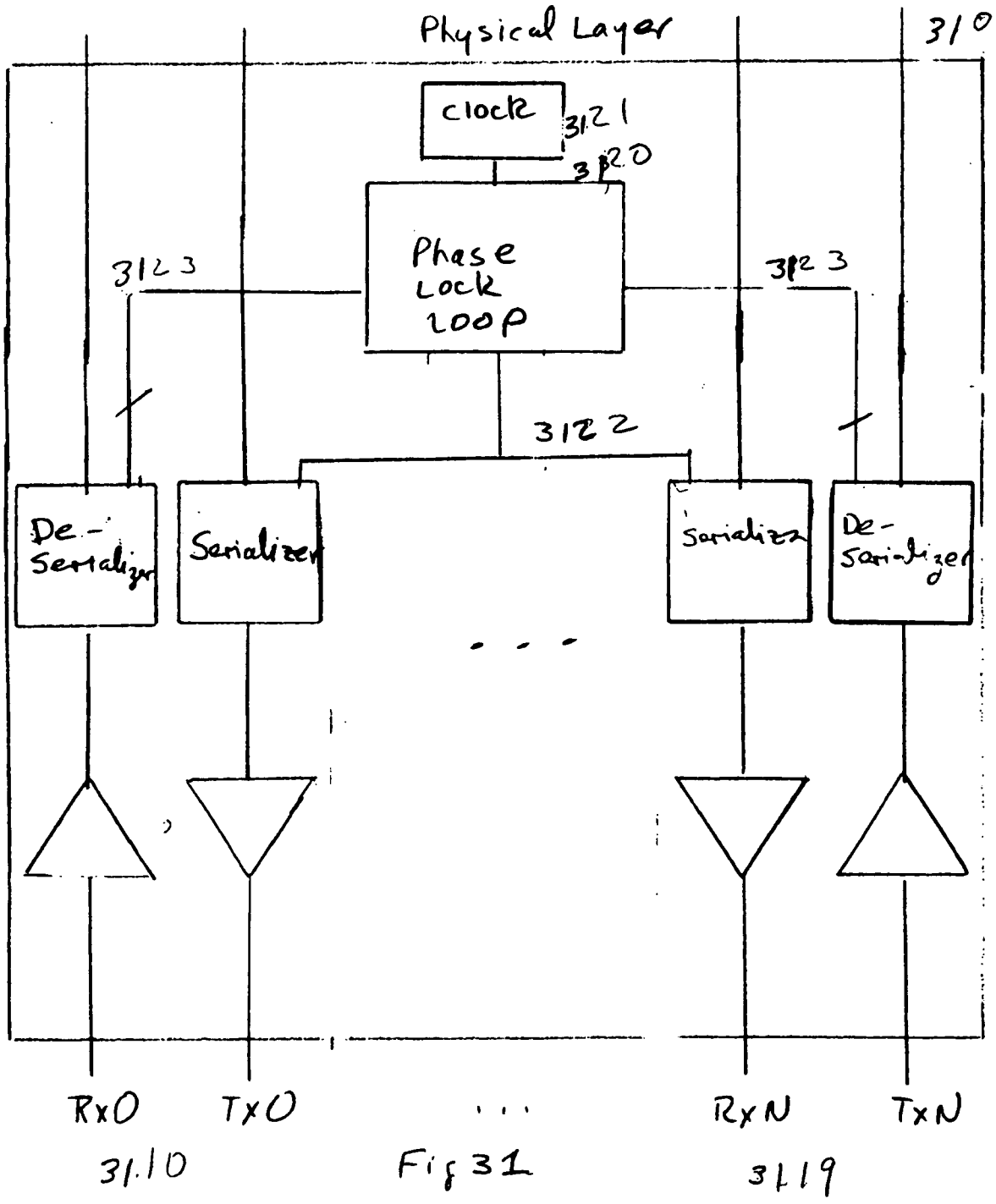


Fig 31

Input Queue 3201

Port	R/W	Address	Data
3	R	1000	
4	W	4000	10....1
3	W	1000	111...0
3	R	2000	
	⋮		

Output Queue 3202

Valid	Port	Data
1	3	11...0
0		
0		
1	3	101...1
	⋮	

Fig 32

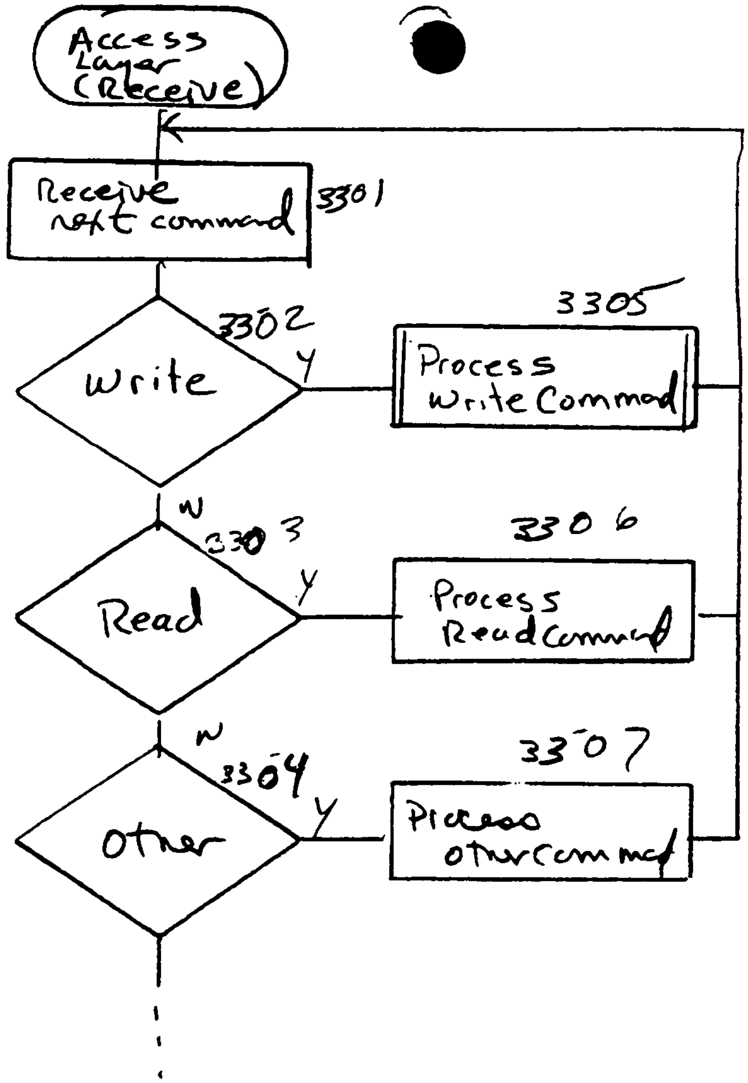


Fig 33

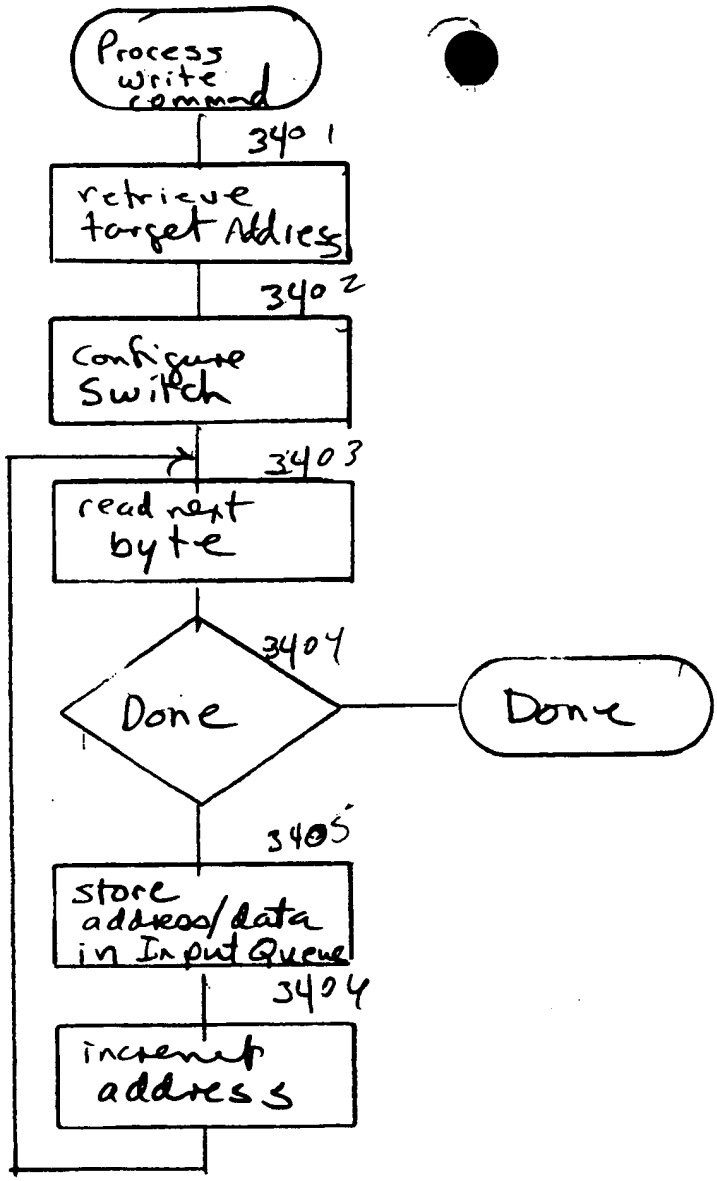

Process write command
3401
retrieve target Address
3402
Configure Switch
3403
read next byte
3404
Done
Done
3405
store address/data in Input Queue
3404
increment address

Fig 34

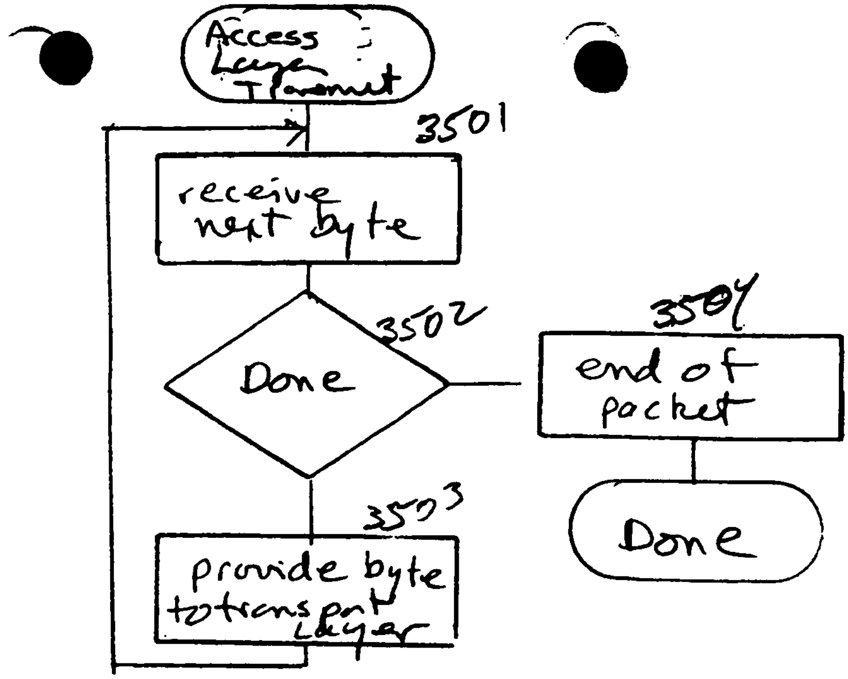


Fig 35

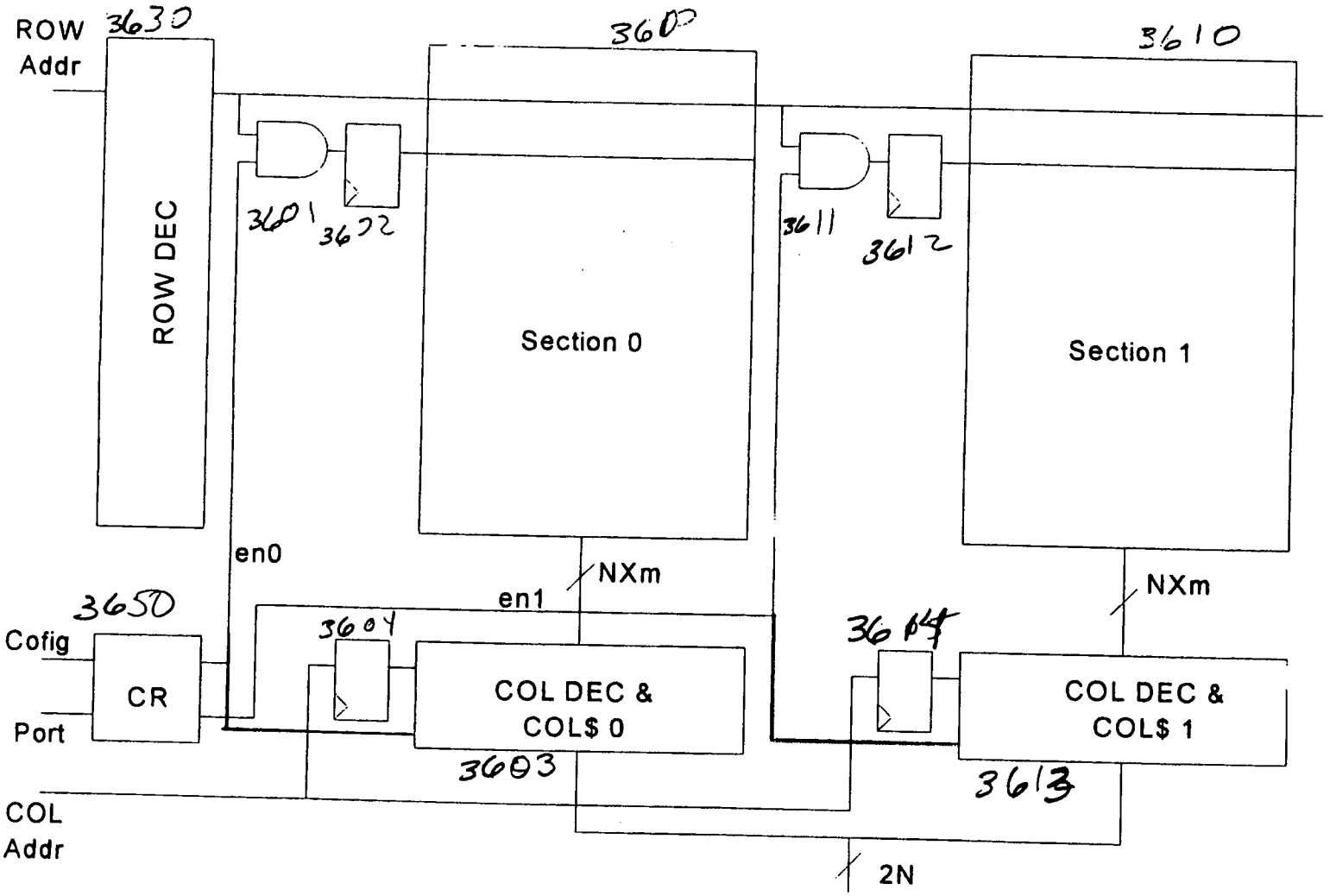
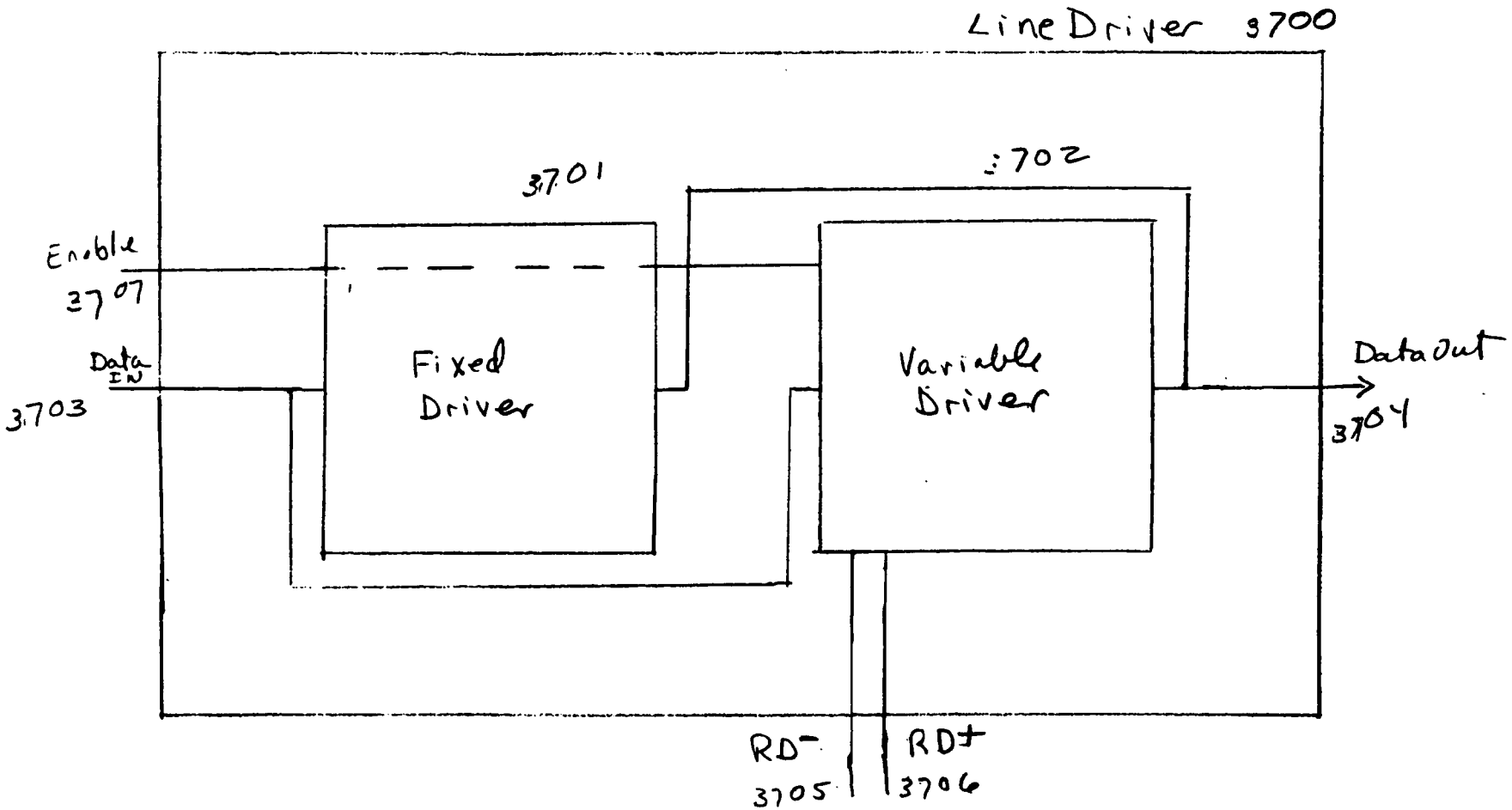


Fig 36



Variable Driver

$\begin{cases} RD^+ \wedge \overline{DataIN} = \text{pull down} \\ RD^- \wedge DataIN = \text{pull up} \end{cases}$

Fig 37A

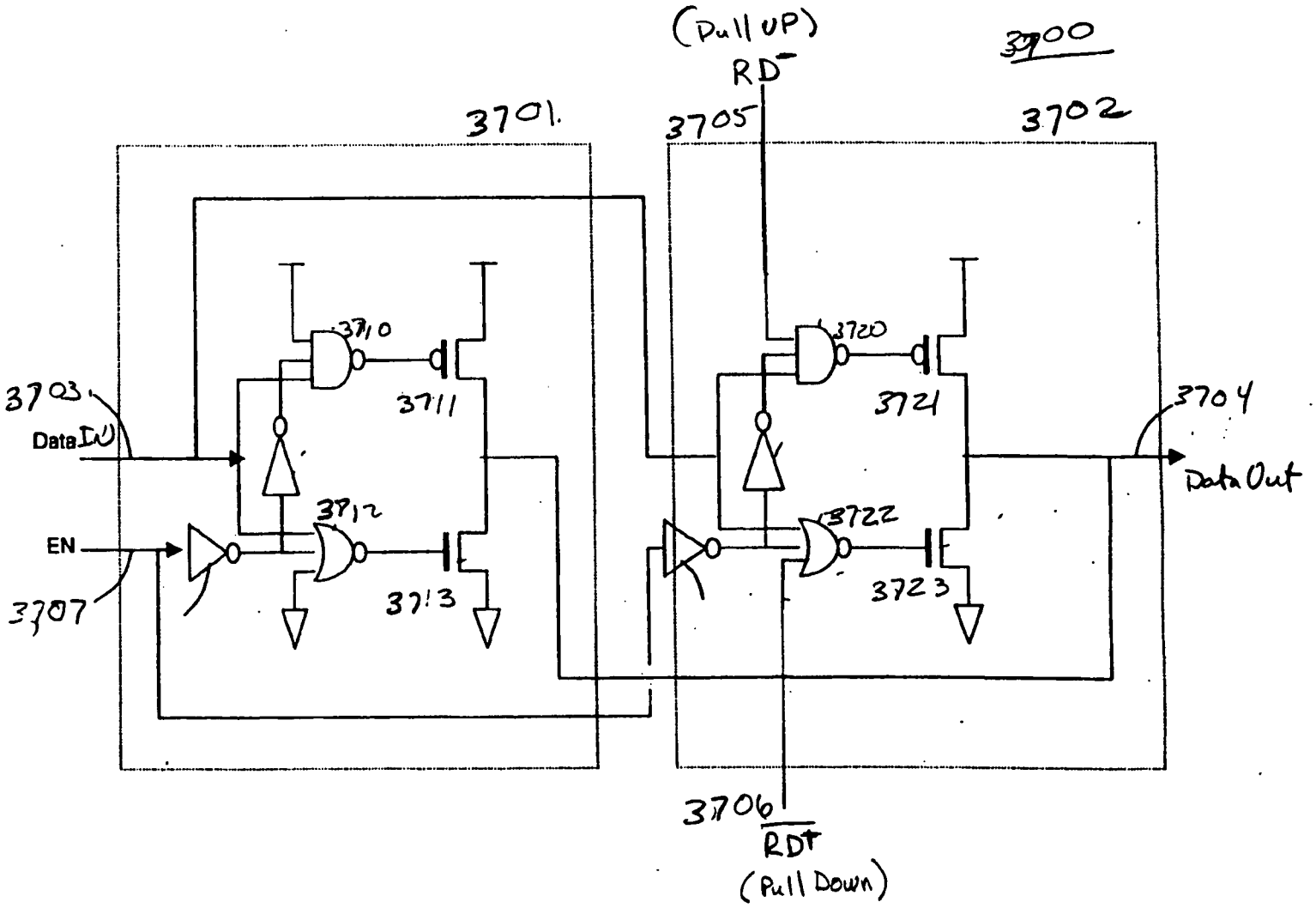


Fig 37B

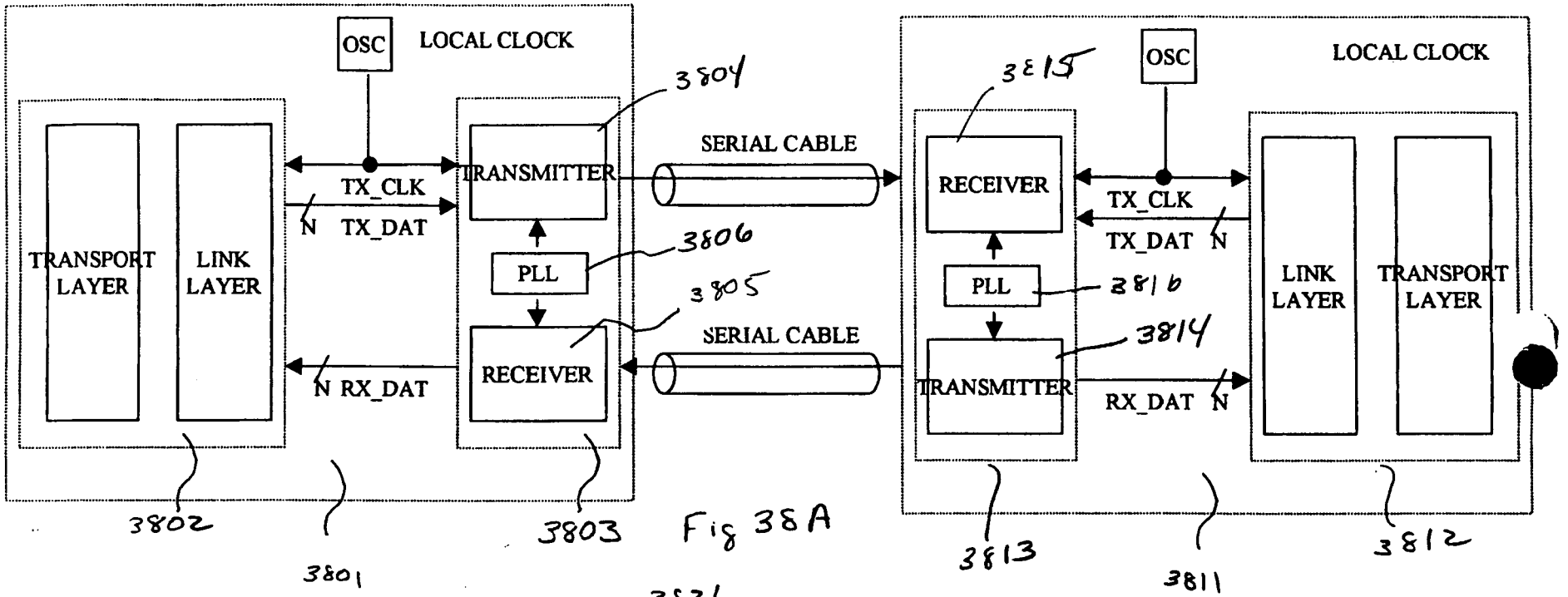


Fig 38A

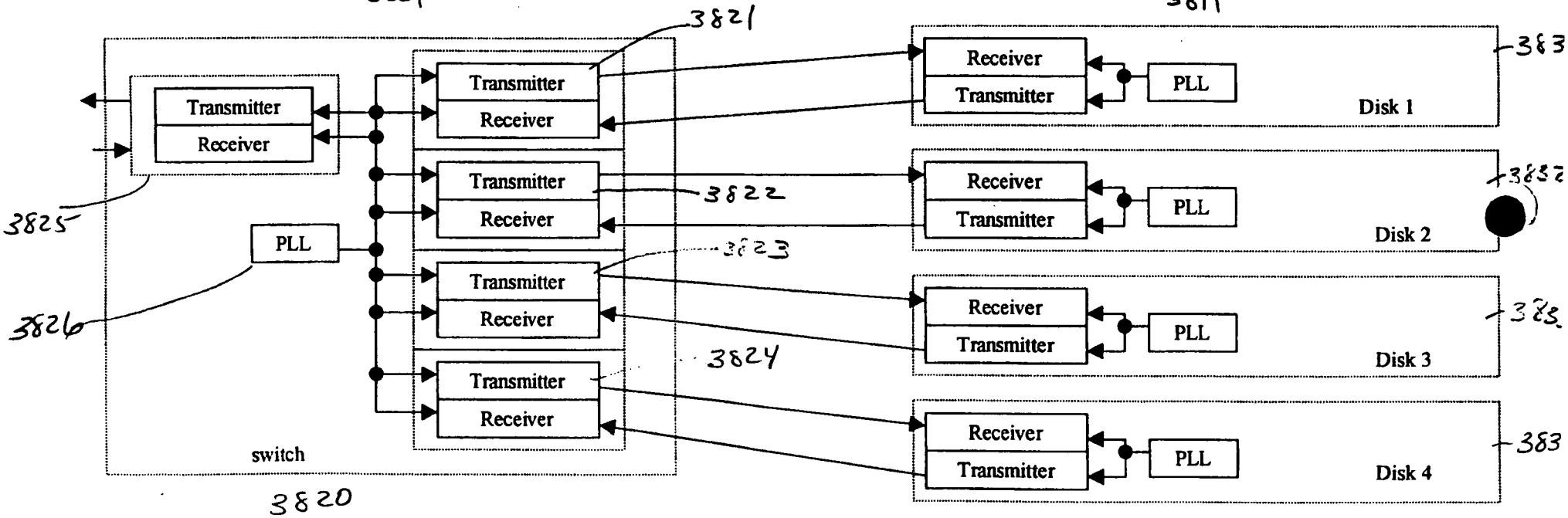


Fig 38B

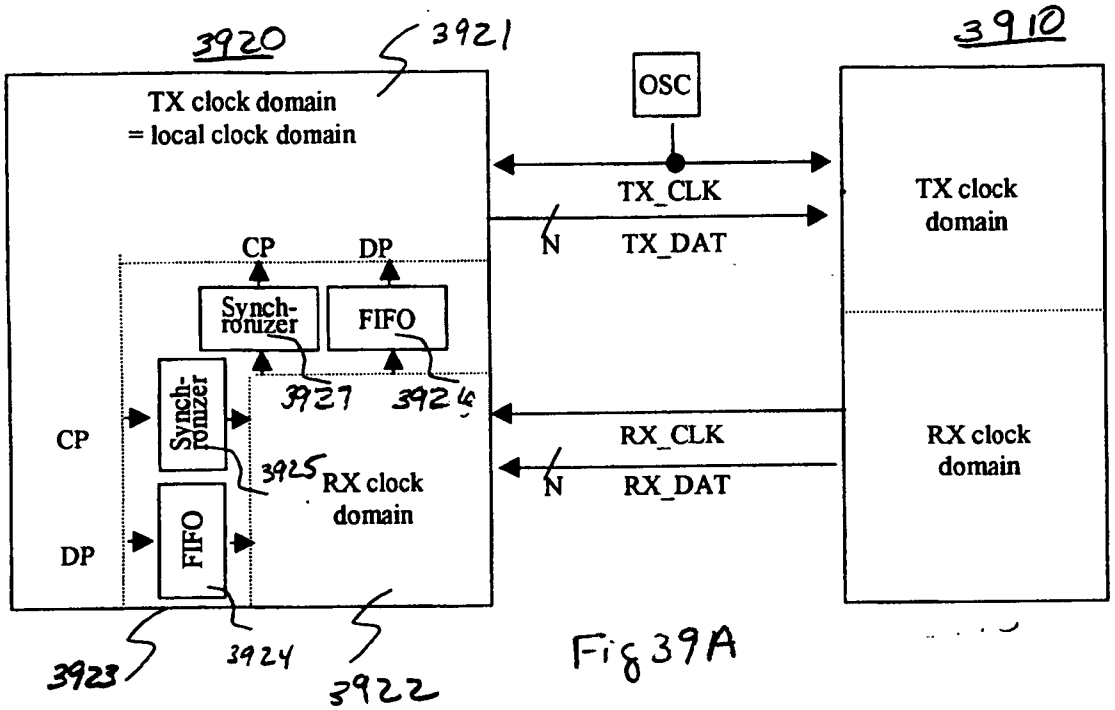


Fig 39A

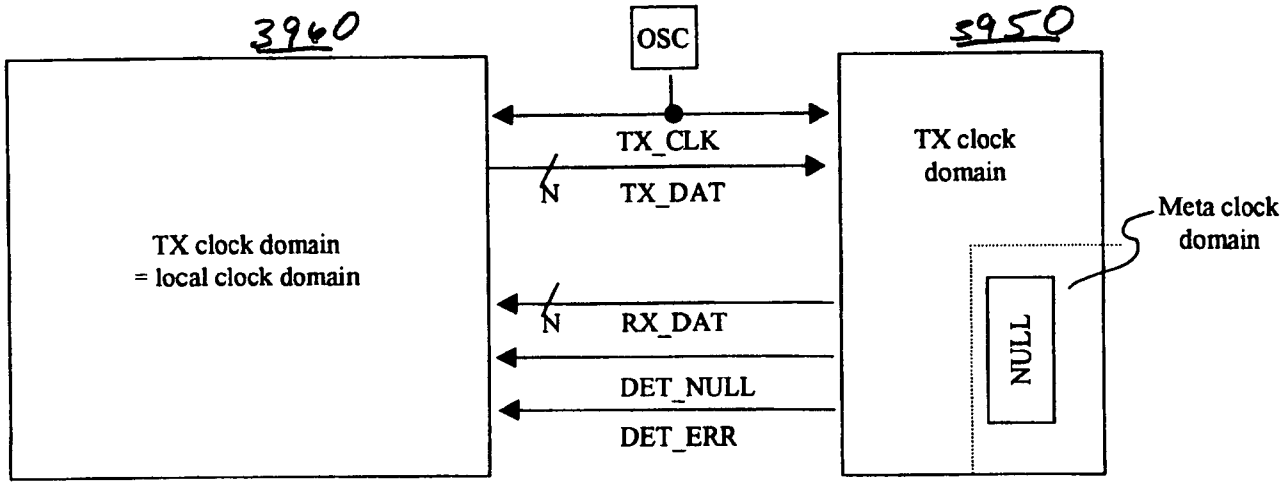


Fig 39B

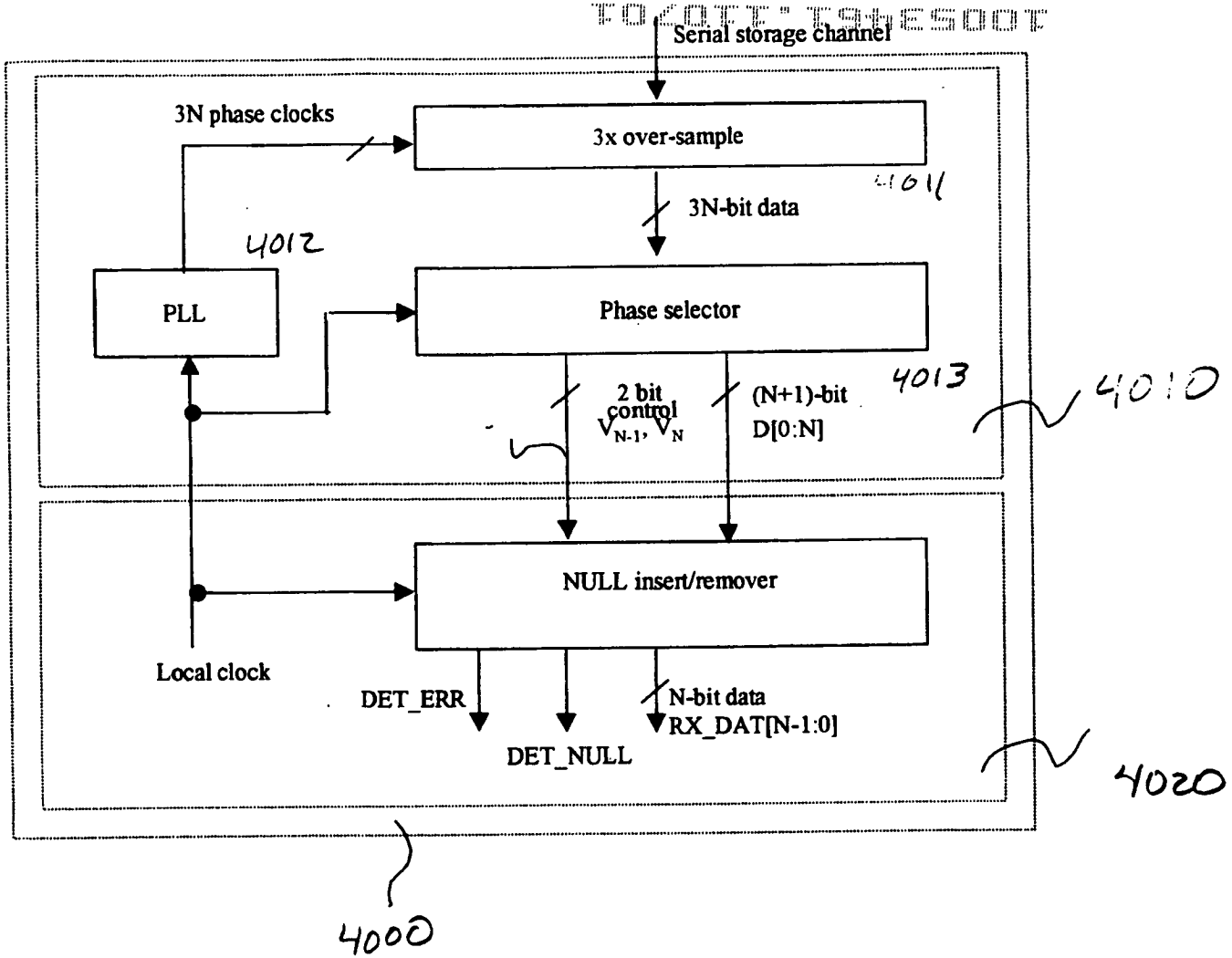

Serial storage channel
3N phase clocks
3x over-sample
4011
3N-bit data
4012
PLL
Phase selector
4013
2 bit control
V_{N-1}, V_N
(N+1)-bit
D[0:N]
4010
NULL insert/remover
Local clock
DET_ERR
N-bit data
RX_DAT[N-1:0]
DET_NULL
4020
4000

Fig 40

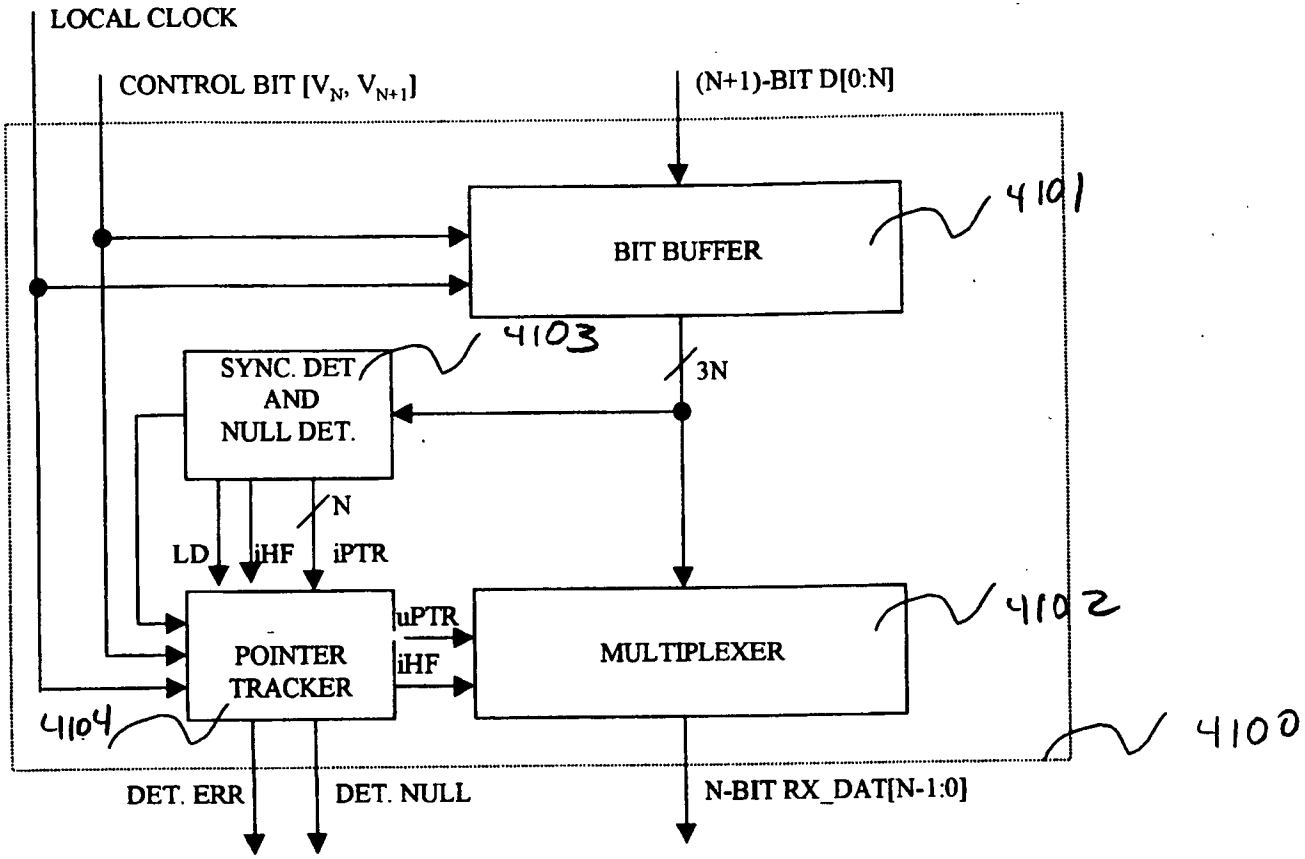


Fig 41

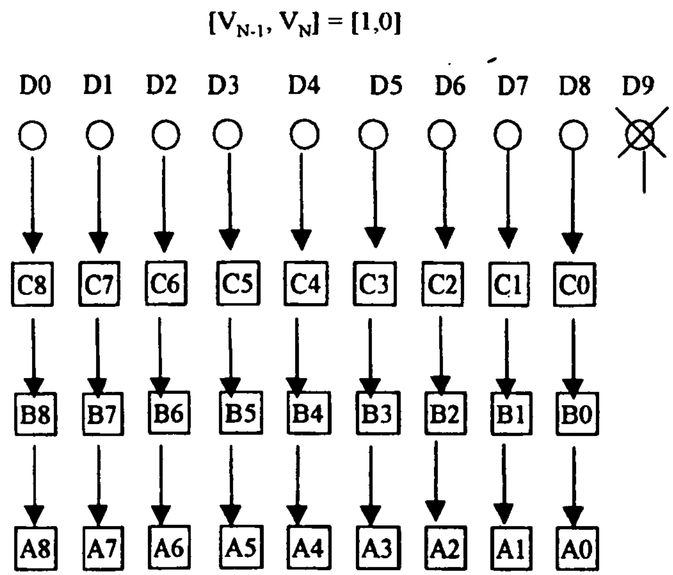

[V_{N-1}, V_N] = [1,0]
D0 D1 D2 D3 D4 D5 D6 D7 D8 D9
C8 C7 C6 C5 C4 C3 C2 C1 C0
B8 B7 B6 B5 B4 B3 B2 B1 B0
A8 A7 A6 A5 A4 A3 A2 A1 A0

Fig 42A

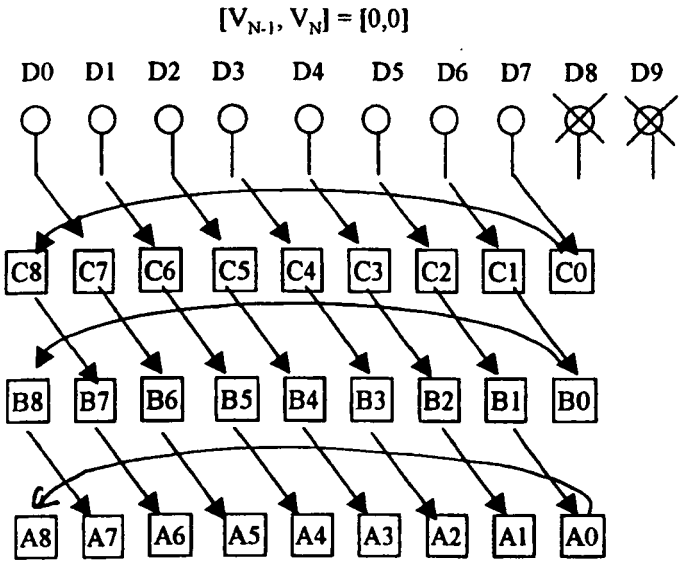


Fig 42B

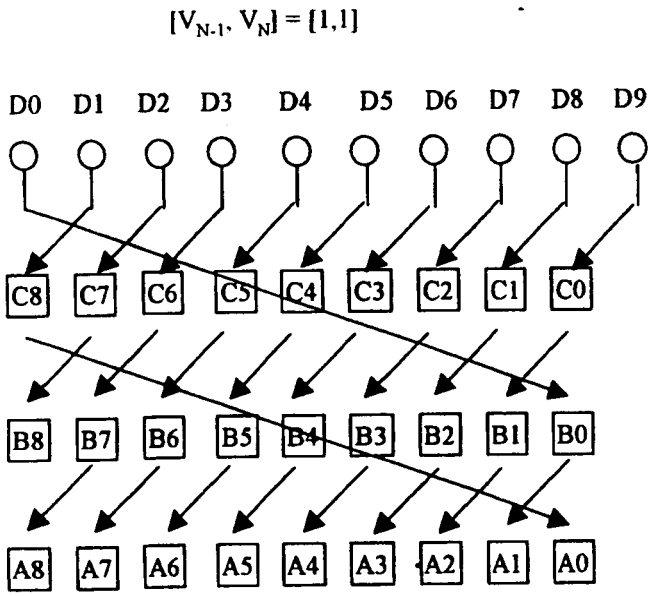


Fig 42C

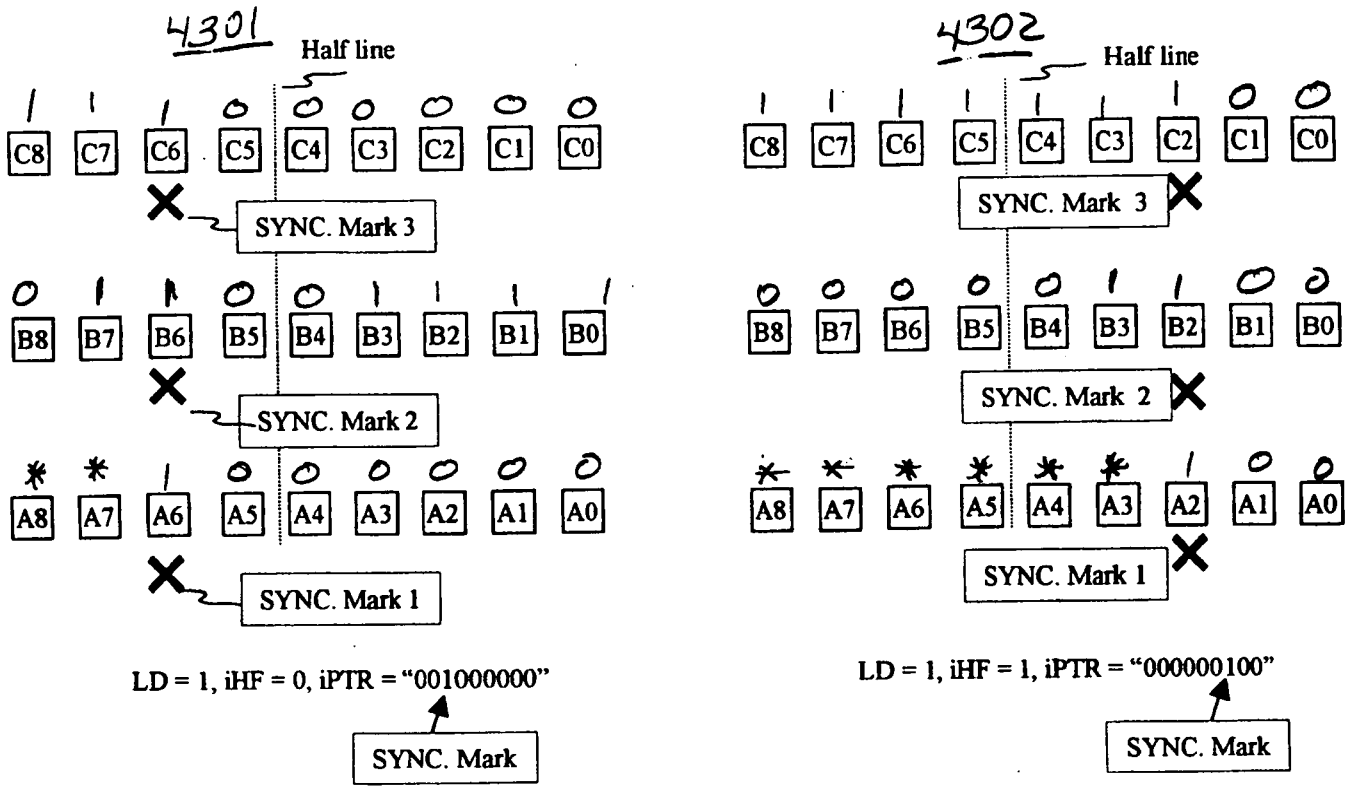
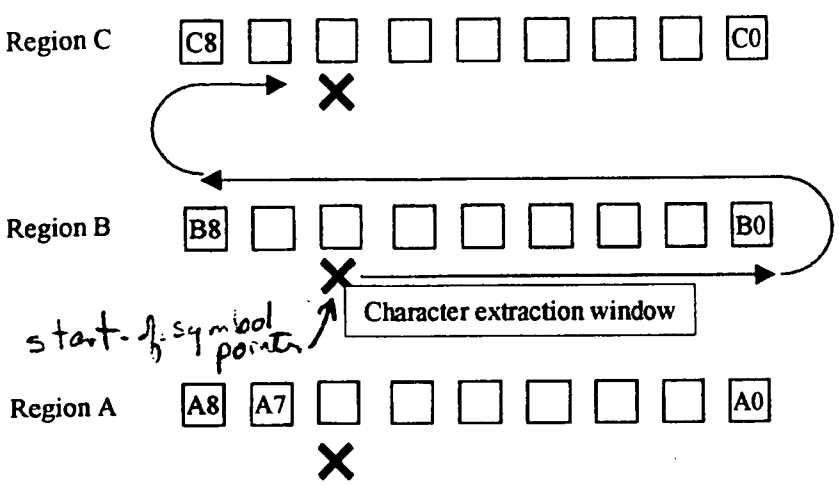
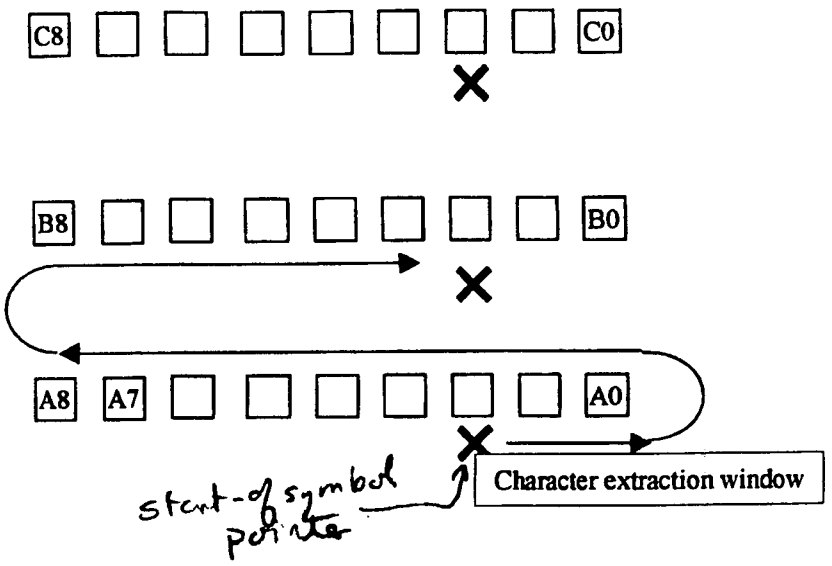


Fig. 43



LD = 1, iHF = 0, iPTR = "001000000"



LD = 1, iHF = 1, iPTR = "000000100"

Fig 44

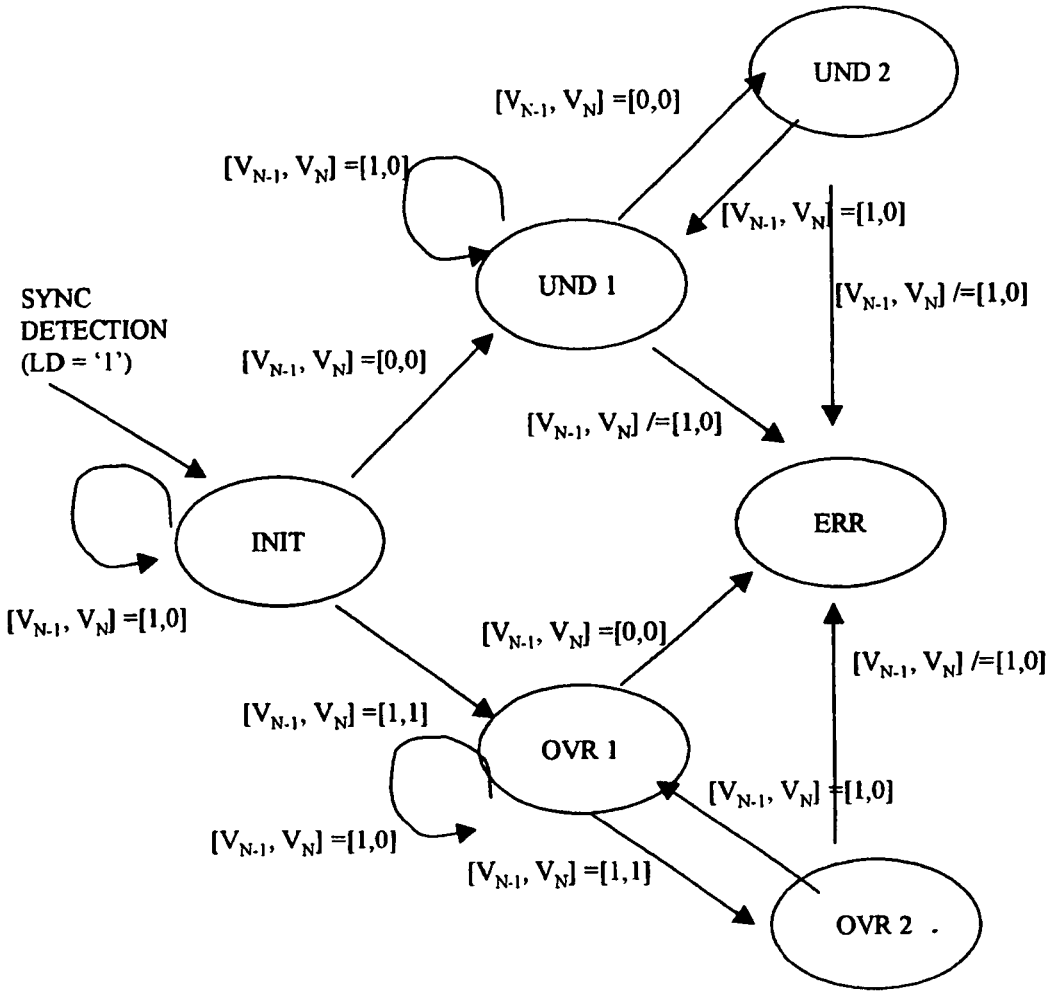

UND 2
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ =[1,0]
UND 1
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ /=[1,0]
SYNC
DETECTION
(LD = '1')
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ /=[1,0]
INIT
ERR
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[0,0]
$[V_{N-1}, V_N]$ /=[1,0]
$[V_{N-1}, V_N]$ =[1,1]
OVR 1
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[1,0]
$[V_{N-1}, V_N]$ =[1,1]
OVR 2

Fig 45

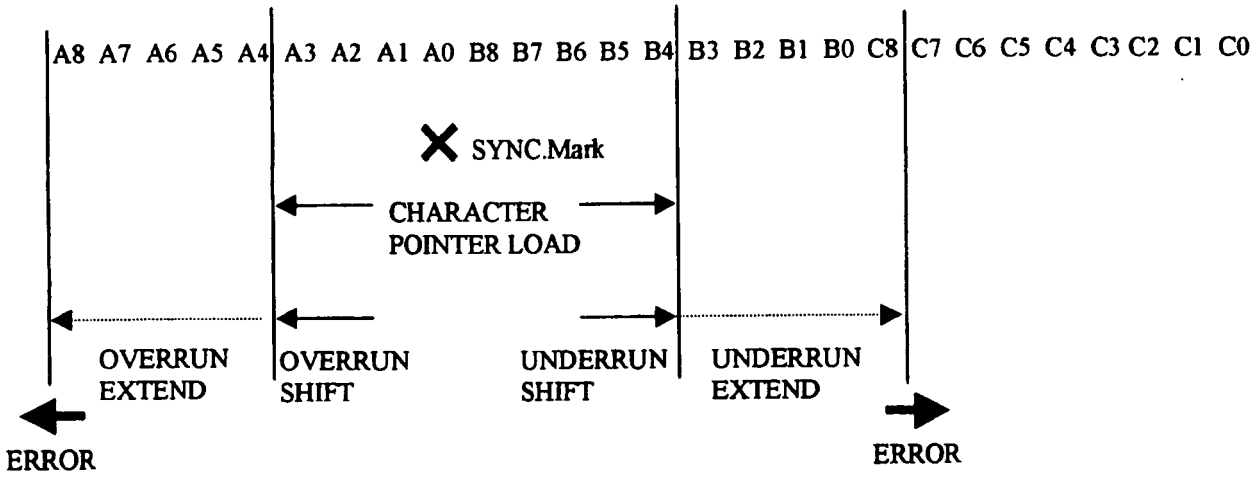


Fig 46

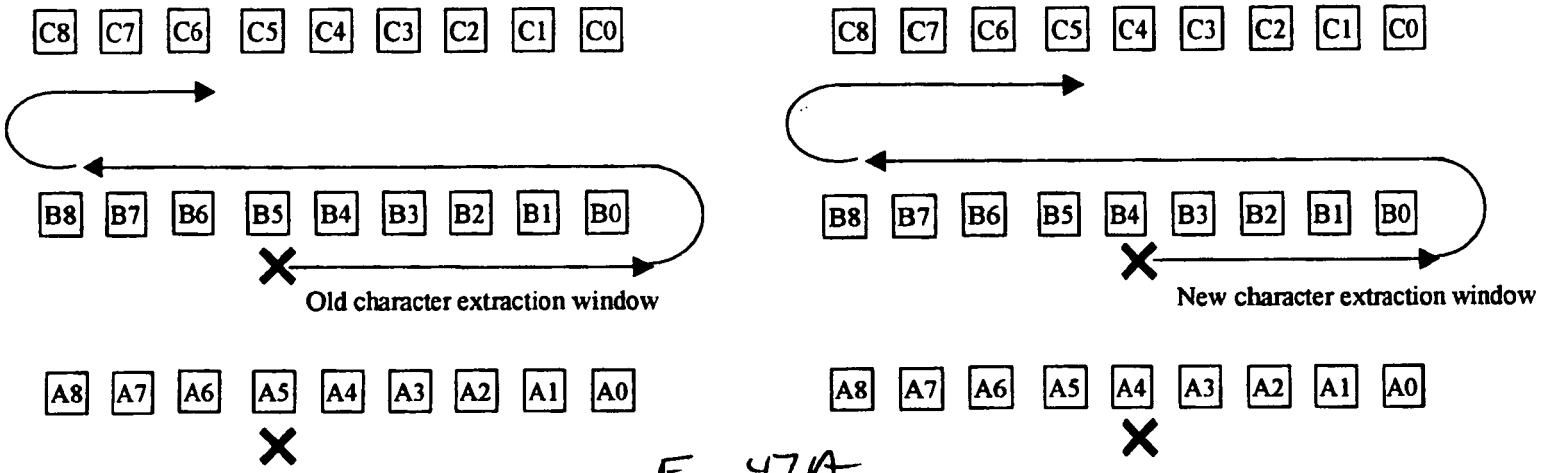


Fig 47A

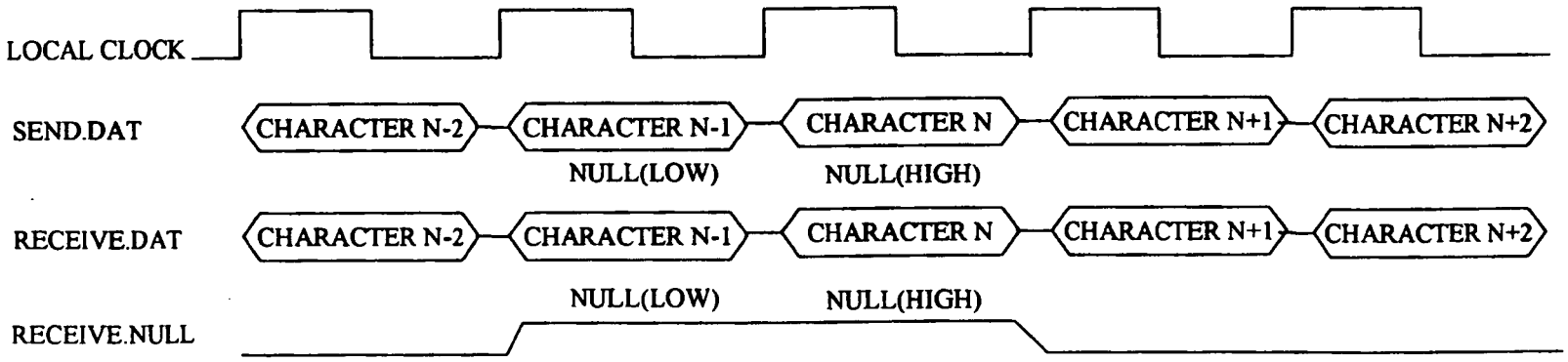


Fig 47B

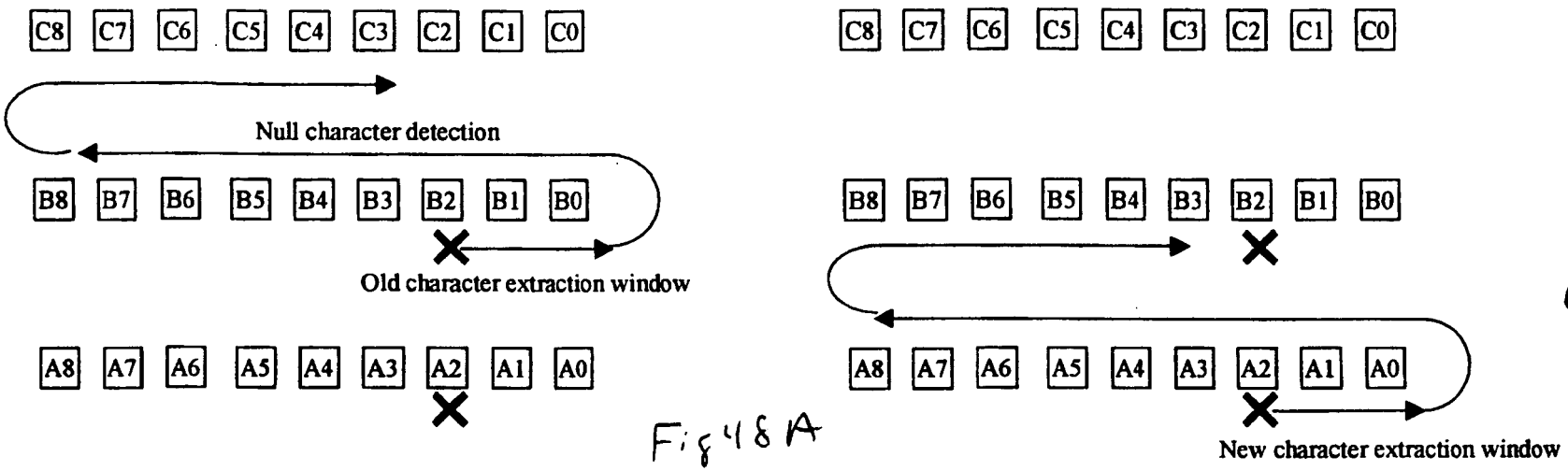


Fig 48A

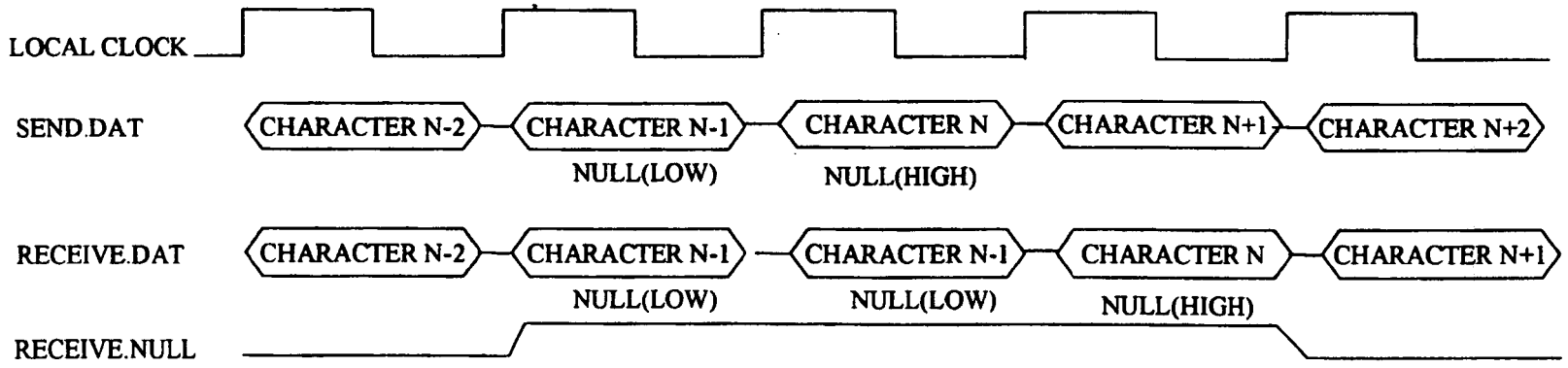


Fig 48B

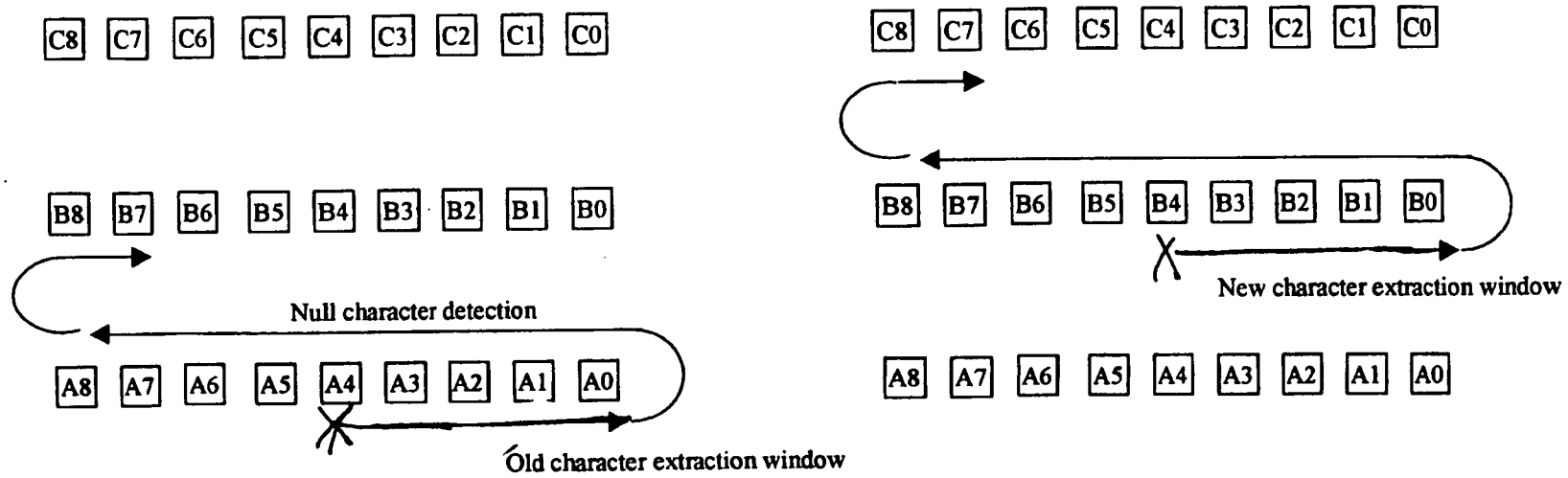


Fig. 49A

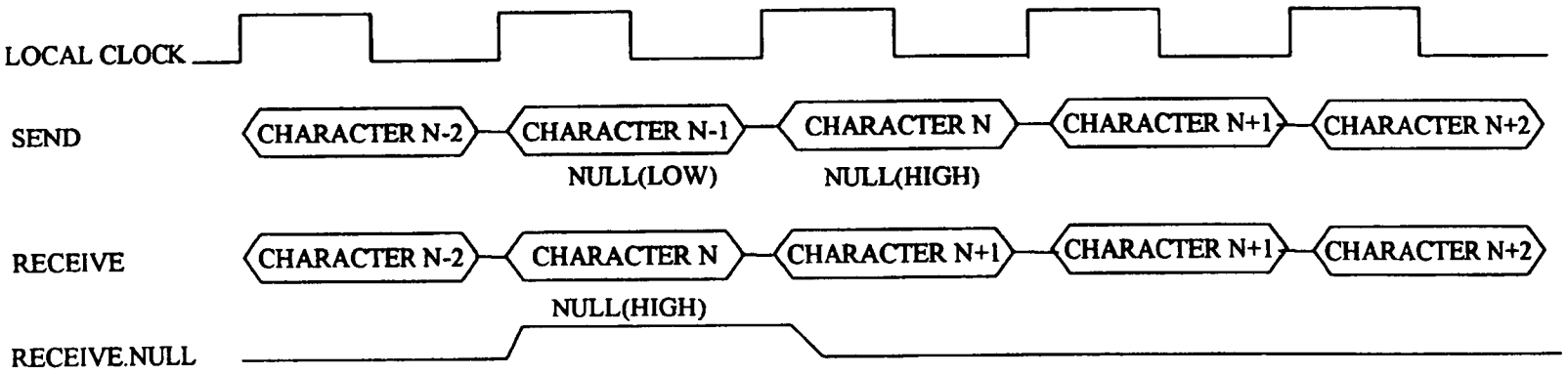


Fig 49B